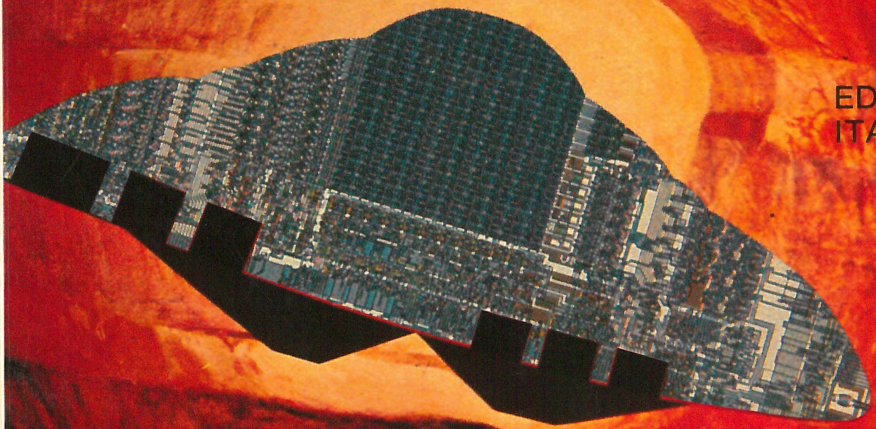
CAPIRE I MICROPROCESSORI

Roland Dubois

EDIZIONE ITALIANA



GRUPPO EDITORIALE JACKSON

P.S.I.

CAPIRE I MICROPROCESSORI

ROLAND DUBOIS

GRUPPO
EDITORIALE
JACKSON
Via Rosellini, 12
20124 Milano

Roland DUBOIS ha 39 anni. È responsabile del dipartimento "sistemi a microprocessori" dell'Efcis, e come tale si occupa della vendita e dell'assistenza tecnica di sistemi e di schede basati su microprocessori. È co-autore del libro "Au coeur des microprocesseurs" (Edizioni Eyrolles), che ha ricevuto nel 1979 il premio dell'Associazione Francese degli Informatici (AFIn). È membro del comitato di redazione della rivista "Minis et Micros".

L'autore ringrazia Roger Carasco, redattore capo aggiunto della rivista "Minis et Micros" per l'aiuto che gli ha dato nella redazione di questo libro.

© Copyright per l'edizione originale P.S.I. Editions du P.S.I. 1980
© Copyright per l'edizione Italiana Gruppo Editoriale Jackson - 1983

Il Gruppo Editoriale Jackson ringrazia per il prezioso lavoro svolto nella stesura dell'edizione italiana la signora Francesca Di Fiore, e l'Ing. Roberto Pancaldi.

Tutti i diritti sono riservati. Stampato in Italia. Nessuna parte di questo libro può essere riprodotta, memorizzata in sistemi di archivio, o trasmessa in qualsiasi forma o mezzo, elettronico, meccanico, fotocopia, registrazione o altri senza la preventiva autorizzazione scritta dell'editore.

Fotocomposizione: Lineacomp S.r.l. - Via Rosellini, 12 - 20124 Milano

Stampato in Italia da:
S.p.A. Alberto Matarelli - Milano - Stabilimento Grafico

PREFAZIONE

Questo libro è, come dice il titolo, una introduzione ai microprocessori. È destinato a tutti coloro che desiderano conoscere i microprocessori e i microcalcolatori.

Questo libro non è un'opera di divulgazione. Spiega in dettaglio, ma in modo abbastanza generale, che cosa è un microprocessore, una memoria ROM, una memoria RAM, un'interfaccia. Mostra come collegare questi diversi circuiti per formare un microcalcolatore. Per rendere più chiara la spiegazione, l'autore ricorre, ogni volta che è necessario, ad esempi pratici: schemi di collegamenti dei diversi circuiti che costituiscono un microcalcolatore, esempi di programmi,...

SOMMARIO

CAPITOLO 1 — INTRODUZIONE AI MICROPROCESSORI 1
— Il microprocessore 1
— Il microcalcolatore 2

CAPITOLO 2 — MICROPROCESSORE, MEMORIE, INTERFACCE 7
— Concetto di bit e di parola 7
— Concetto di decodificatore 7
— Concetto di registro 8
— I bus e i segnali di comando di un microprocessore 8
— Organizzazione di un microprocessore 12
— Organizzazione di una memoria RAM 17
— Organizzazione di una memoria ROM 19
— Organizzazione di un'interfaccia 21
— Indirizzamento delle interfacce e dei circuiti di memoria 22
— Inizializzazione di un'interfaccia 24

CAPITOLO 3 — I DIVERSI TIPI DI INTERFACCIA 27
— INTERFACCIA PARALLELA 27
● Collegamento di un microprocessore ad un plotter 29
● Collegamento di un microprocessore ad una tastiera 31
● Collegamento di un microprocessore ad un display a 7 segmenti 32
— INTERFACCIA SERIE 34
● Organizzazione esterna 35
● Organizzazione interna 36
● Comunicazione serie asincrona 36
● Comunicazione serie sincrona 37
● Interfacce serie specializzate 39
— INTERFACCIA DI UNA UNITA' A FLOPPY DISK 41
● Come "formattare" il floppy-disk 41
● Organizzazione esterna dell'interfaccia 45
● Organizzazione interna dell'interfaccia 46
● I comandi dell'unità a floppy-disk 47
— CIRCUITO DI ACCESSO DIRETTO ALLA MEMORIA (DMA) 47
● Inizializzazione di un circuito di DMA 51
— TEMPORIZZATORE/CONTATORE DI EVENTI 51
— INTERFACCIA DI UN VIDEO 53
● Principio di funzionamento 54
● Organizzazione esterna e interna 56

CAPITOLO 4 — I MODI DI INDIRIZZAMENTO 61
— INDIRIZZAMENTO ESTESO 61
— INDIRIZZAMENTO INDIRETTO TRAMITE EGISTRI 62
— INDIRIZZAMENTO INDICIZZATO 63
● Indirizzamento indicizzato semplice 63
● Indirizzamento indiretto indicizzato 63
● Indirizzamento indiretto indicizzato con "spiazzamento" 64
— INDIRIZZAMENTO RELATIVO 65
● Indirizzamento relativo lungo 65
● Indirizzamento relativo corto 65
— INDIRIZZAMENTO IMPLICITO 67
— INDIRIZZAMENTO IMMEDIATO 67

CAPITOLO 5 — SET DI ISTRUZIONI 69
— ISTRUZIONI DI TRASFERIMENTO E DI SCAMBIO 69
— ISTRUZIONI ARITMETICHE 72
— ISTRUZIONI LOGICHE 73
— ISTRUZIONI DI ROTAZIONE E DI SHIFT 73
— ISTRUZIONI A LIVELLO DI BIT 73
— ISTRUZIONI DI MANIPOLAZIONI DI CARATTERI 75
— SALTI E RITORNI AL PROGRAMMA PRINCIPALE 75
— ISTRUZIONI DI INGRESSO/USCITA 75
— ISTRUZIONI DI CONTROLLO DELL'UNITA' CENTRALE 76
— ESEMPI DI PROGRAMMAZIONE 77
● Addizione decimale di due numeri di tre cifre 77
● Shift di una cifra a destra (microprocessore a 8 bit) 78
● Trasferimento di un blocco di dati (microprocessore a 8 bit) .. 79
● Ricerca di un carattere in un blocco di dati (microprocessore a 8 bit) 80

CAPITOLO 6 — I SISTEMI DI SVILUPPO 83
— LINGUAGGIO ESADECIMALE 83
— LINGUAGGIO ASSEMBLER 84
— LINGUAGGI EVOLUTI 87

CAPITOLO 7 — DESCRIZIONE DEI PRINCIPALI MICROPROCESSORI ESISTENTI SUL MERCATO 91
— MOTOROLA MC 6800 92
— MOTOROLA MC 6809 94
— MCS 6502 A TECNOLOGIA MOS 96
— INTEL 8080 A 98
— ZILOG Z80 100
— SIGNETICS 2650 102

— RCA CDP 1802 104
— SC/MP II DELLA NATIONAL SEMICONDUCTOR 106
— FAIRCHILD F8 108
— INTERSIL IM 6100 110
— TEXAS TMS 9900 112
— INTEL 8086 114
— ZILOG Z 8000 117
— MOTOROLA MC 68000 120
— NS 16000 122

CAPITOLO 8 — PASSATO, PRESENTE E FUTURO 125
— I LIMITI DELLA TECNOLOGIA VLSI 125
— LA SITUAZIONE ATTUALE 126
— IL FUTURO DEL MERCATO VLSI 126

CAPITOLO 1

INTRODUZIONE AI MICROPROCESSORI

Il microprocessore, nato nel 1972, è il risultato dell'evoluzione tecnologica da una parte e delle esigenze del mercato dall'altra.

Lo sviluppo della tecnologia dei semiconduttori, negli anni '70, era arrivato ad un livello tale da rendere possibile la fabbricazione di circuiti con diverse migliaia di transistor, che realizzavano funzioni sempre più complesse.

Ma il costo della realizzazione di simili circuiti poteva essere ammortizzato solo da una produzione su larghissima scala; di conseguenza, le case costruttrici di componenti elettronici furono spinte a realizzare circuiti che si adattassero al più grande numero di applicazioni possibile, pur restando utilizzabili in piccola serie da un certo numero di utenti. Il risultato di questa tendenza fu un circuito logico programmabile dall'utente, che prese il nome di "microprocessore".

IL MICROPROCESSORE

Questo circuito possiede due caratteristiche fondamentali:

— è un elaboratore, cioè un dispositivo in grado di elaborare le informazioni;
— è un circuito integrato, cioè un insieme indivisibile di transistor, su un unico supporto, che realizza diverse funzioni.

I microprocessori esistono solo dal 1972, ma la loro architettura non è nuova, e si ispira ai concetti generali dell'architettura dei grandi sistemi informatici.

Un microprocessore realizza operazioni aritmetiche (addizione, sottrazione, moltiplicazione, divisione) e operazioni logiche (AND, OR, EXCLUSIVE-OR). Effettua shift e rotazioni a destra o a sinistra su numeri binari, confronti, maschere, etc.. Prende decisioni a seconda che il risultato di un'operazione sia superiore, inferiore o uguale a un certo dato, o a seconda che sia positivo o negativo. In breve, si comporta esattamente come l'unità centrale di un elaboratore o di un minielaboratore, ma se ne differenzia in due caratteristiche:

— è in genere, più lento (1), e può essere quindi utilizzato solo in certe applicazioni;
— costa di meno.

Il microprocessore è un circuito integrato, e come tale presenta le caratteristiche della tecnologia a semiconduttori: ha dimensioni ridotte, e contiene, su di un chip di silicio di qualche millimetro quadrato, diverse decine di migliaia di transistor. Presenta naturalmente anche i limiti di questa tecnologia. Il primo consiste nella difficoltà di ottenere un buon rendimento in fase di fabbricazione. Un aumento del 10% nelle dimensioni del chip può diminuire questo rendimento del 50%.

Ricordiamo peraltro che la complessità dei circuiti è raddoppiata ogni anno dalla comparsa del primo circuito integrato.

Il supporto in cui è contenuto il microprocessore rappresenta un secondo limite, in quanto non può avere più di un certo numero di piedini. Inoltre è un elemento che pesa considerevolmente nel costo di fabbricazione dei circuiti integrati; e le case costruttrici di microprocessori, se vogliono esser competitive, devono utilizzare dei supporti standard.

Il numero ridotto di piedini (40 o 64 al massimo per gli "standard" attuali) limita le possibilità di invenzione dei progettisti di microprocessori.

Un ultimo limite è dato dalla potenza che può essere dissipata dal supporto, che, se realizzato in plastica, non può dissipare più di 700 mW; solo la tecnologia MOS consente di costruire un microprocessore completo senza superare questo livello di dissipazione.

IL MICRO-CALCOLATORE

Un microprocessore da solo non sa fare niente. Perchè possa lavorare occorre collegargli delle memorie e delle interfaccie. Una volta collegato a questi circuiti, il microprocessore diventa allora un micro-calcolatore(2) che comprende quattro parti (fig. 1) che comunicano tramite un bus dati, un bus di indirizzo e un bus di controllo. Queste quattro parti sono:
— il microprocessore;
— la memoria di programma, che segnala al microprocessore le funzioni che deve realizzare;
— la memoria dati, che immagazzina i dati provenienti dalle periferiche, i risultati intermedi e i risultati finali;

---

(1) I nuovi microprocessori a 16 bit a tecnologia H-MOS hanno un'architettura e una velocità tali da consentir loro di competere con i minielaboratori. Sicuramente questi nuovi circuiti finiranno per invadere il mercato tradizionalmente riservato ai "mini".

(2) Esistono attualmente dei micro-calcolatori realizzati su di un solo chip di silicio. Si chiamano "single chip"; hanno una capacità limitata di memoria e possono collegarsi al massimo ad una trentina di dispositivi di ingresso/uscita.

— le interfaccie, che consentono il trasferimento dei dati dalle periferiche verso il microprocessore e viceversa.

Il microprocessore funziona sotto il controllo di un programma contenuto nella memoria di programma.

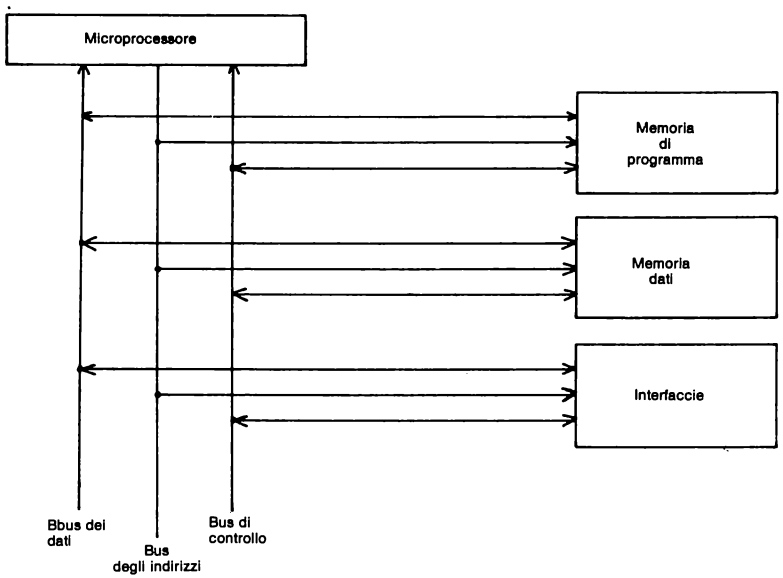


Figura 1 - *Schema generale di un microcalcolatore.*

Un programma è una sequenza di istruzioni che il microprocessore deve eseguire. Diversamente dagli elaboratori, che costano molto ma consentono la gestione di più applicazioni simultaneamente, il microprocessore è relativamente economico ma viene in generale utilizzato per un'unica applicazione.

Il programma che corrisponde a questa applicazione, è contenuto in una memoria a sola lettura, che peraltro può essere letta tutte le volte che lo si desidera. Essa mantiene il suo contenuto anche dopo che l'alimentazione è stata tolta. Non è necessario, come nel caso di un elaboratore, caricare il programma in memoria ogni volta che la macchina viene accesa.

Esistono diversi tipi di memoria a sola lettura:

— le ROM (Read Only Memory), in cui viene scritto dalla casa costruttrice il programma fornito dall'utente;
— le PROM, che sono direttamente programmabili dall'utente;
— le REPROM, che sono programmabili dall'utente, e riprogrammabili dopo essere state cancellate.

Per quei lettori che non conoscono le memorie a semiconduttore, ci spieghiamo con un'analogia: una ROM è come un libro che può essere letto tutte le volte che si vuole, ma che non può essere modificato in alcun modo. Una PROM è come un foglio di carta su cui l'utente può scrivere quello che vuole; se si sbaglia, ricomincia su un altro foglio. Infine, una REPROM è come una lavagna, su cui si può scrivere e cancellare a volontà.

Per eseguire il suo programma, il microprocessore utilizza dei dati, che sono contenuti nella memoria dati, o che provengono dalle periferiche tramite le interfaccie.

La memoria dati è notevolmente diversa dalla memoria del programma. È una memoria "viva" (RAM, Random Access Memory), in cui l'utente può scrivere e leggere a volontà.

Le interfaccie sono dei circuiti di ingresso/uscita, la cui funzione è definita da programma. Essi consentono la comunicazione tra il microprocessore e le periferiche.

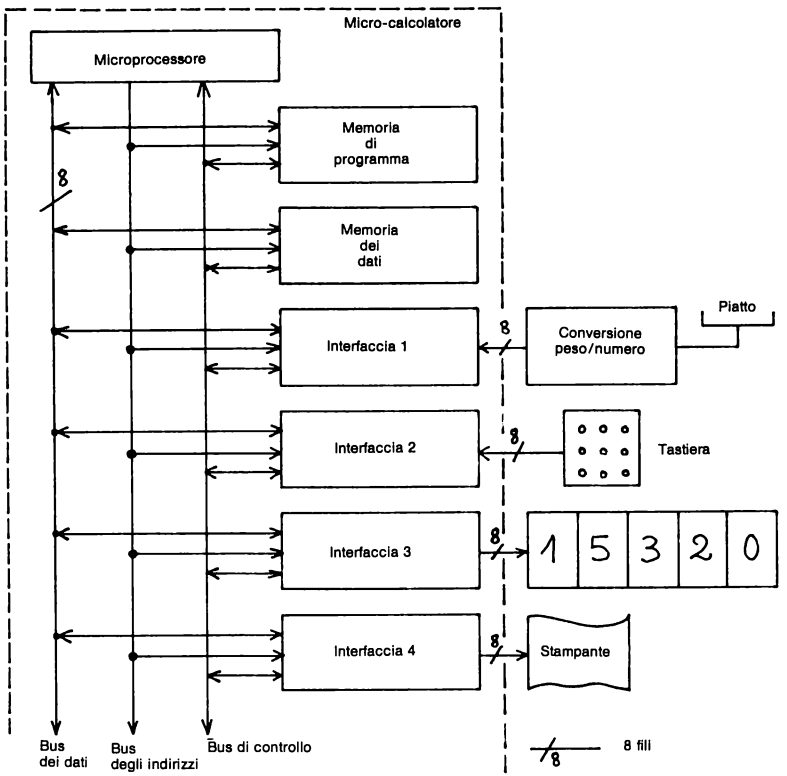


Figura 2 - *Schema di principio di una bilancia.*

Per illustrare come funziona un microprocessore, prendiamo l'esempio della bilancia elettronica (figura 2).

Questa bilancia comprende, oltre al microcalcolatore, diverse periferiche: il piatto su cui si posa la merce; dei display numerici che visualizzano le diverse informazioni sul peso e sul prezzo; una stampante che copia i vari prezzi e i risultati totali.

Le comunicazioni tra il microprocessore e le diverse periferiche sono rese possibili da interfaccie parallele (cioè circuiti che hanno 8 linee verso le periferiche, programmabili come ingressi o come uscite). L'interfaccia 1 che collega il piatto al microprocessore è programmata come ingresso, ma il collegamento non è diretto, in quanto è necessario un sistema di conversione; in effetti i pesi devono essere trasformati in numeri, per essere elaborati dal microprocessore. L'interfaccia 2, che collega la tastiera al microprocessore, è anch'essa programmata come ingresso, mentre quelle che collegano il microprocessore ai display e alla stampante sono programmate come uscite. Notate il significato delle frecce nella figura.

Vediamo ora come funziona la bilancia. Il venditore posa la prima merce del cliente sul piatto. Il microprocessore, sotto il controllo del programma, legge il peso della merce, lo registra nella memoria dati, lo visualizza sul display e lo stampa. Il negoziante inserisce allora il prezzo per kg. della merce. Il microprocessore, sempre sotto il controllo del programma, registra il prezzo nella memoria dati, lo visualizza e lo stampa. Poi esegue la moltiplicazione del peso della merce per il prezzo di un chilo, registra il risultato in memoria la visualizza e la stampa. A questo punto il negoziante posa la seconda mere del cliente sul piatto, e si ripetono le operazioni precedenti; così di seguito, fino a quando tutte le merci del cliente sono state pesate. Alla fine il negoziante schiaccia il tasto "TOTALE", e il microprocessore esegue la somma dei diversi prezzi delle merci, la visualizza e la stampa. Il cliente paga e l'operazione precedente ricomincia con un altro cliente, e così di seguito.

Notiamo che la memoria dati funziona da memoria temporanea. Conserva i pesi e i prezzi delle merci di un cliente; una volta che questi è stato servito, le informazioni relative sono cancellate da quelle del nuovo cliente.

Il programma si compone di tre parti: inizializzazione del sistema, elaborazione propriamente detta della pesata, e calcolo del totale.

L'inizializzazione del sistema avviene al momento della accensione della bilancia, e consiste nella programmazione delle interfaccie come ingressi o uscite.

L'elaborazione della pesata ha luogo ogni volta che il negoziante pone una merce sul piatto della bilancia.

Il calcolo del totale viene eseguito ogni volta che il negoziante schiaccia il tasto "TOTALE".

Il programma è naturalmente molto più complesso di quello che è stato descritto prima. La figura 3 descrive in modo chiaro e comprensibile i diversi comandi o istruzioni che il microprocessore deve eseguire. Le istruzioni sono scritte nel linguaggio umano per fare capire ai lettori che non conoscono la programmazione che cos'è un programma e che cos'è un'istruzione.

INIZIALIZZAZIONE

INTERFACCIA 1 "INGRESSO"
INTERFACCIA 2 "INGRESSO"
INTERFACCIA 3 "USCITA"
INTERFACCIA 4 "USCITA"

PESATA

LEGGERE INTERFACCIA 1 (PESO DELLA MERCE)
REGISTRARE NELLA MEMORIA
INVIARE DATI ALL'INTERFACCIA 3 (DISPLAY)
INVIARE DATI ALL'INTERFACCIA 4 (STAMPANTE)
LEGGERE INTERFACCIA 2 (PREZZO AL KG.)
REGISTRARE NELLA MEMORIA
INVIARE DATI ALL'INTERFACCIA 3 (DISPLAY)
INVIARE DATI ALL'INTERFACCIA 4 (STAMPANTE)
MOLTIPLICARE IL PESO PER IL PREZZO
REGISTRARE NELLA MEMORIA
INVIARE DATI ALL'INTERFACCIA 3 (DISPLAY)
INVIARE DATI ALL'INTERFACCIA 4 (STAMPANTE)

TOTALE

SOMMA DEI DIVERSI PREZZI DELLE MERCI DELLA CLIENTE

Figura 3 - *Programma del microprocessore di una bilancia.*

CAPITOLO 2

MICROPROCESSORI, MEMORIE, INTERFACCE

Nel primo capitolo abbiamo presentato il microcalcolatore nel suo complesso. In questo capitolo esamineremo l'organizzazione esterna e interna dei diversi circuiti che costituiscono un microcalcolatore.

CONCETTO DI BIT E DI PAROLA

Il microprocessore conosce solo il linguaggio binario, che è composto da parole e numeri basati su un elemento di informazione che si chiama bit (binary digit). Il bit è caratterizzato da due stati (0 e 1) che corrispondono nel microprocessore a due tensioni, in genere 0 e 5 Volt.

L'insieme di più bit consente di distinguere più di due stati. Per esempio, con una parola di tre bit è possibile identificare 2^3 (cioè 8) stati, cioè di contare da 0 a 7 (figura 1).

Con una parola di 8 bit, che prende il nome di byte, si possono distinguere 2^8 stati. Si può quindi scegliere una istruzione tra 256, oppure indirizzare 256 posizioni di memoria, oppure contare da 0 a 255.

CONCETTO DI DECODIFICATORE

Un decodificatore identifica i diversi stati che assume una parola binaria. Per esempio, un decodificatore a 3 bit (figura 2) consente il riconoscimento di 8 stati.

Parola	Stati
0 0 0	0
0 0 1	1
0 1 0	2
0 1 1	3
1 0 0	4
1 0 1	5
1 1 0	6
1 1 1	7

Figura 1 - *Diversi stati corrispondenti a una parola di tre bit.*

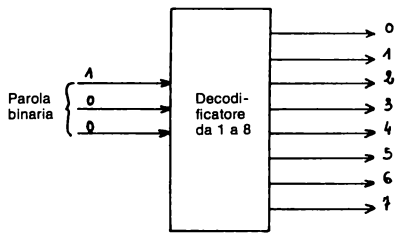


Figura 2 - *Decodificatore da 1 a 8. A seconda dello stato dei piedini di ingresso (in questo caso 100), viene selezionata una delle linee di uscita (qui la linea 4, che corrisponde alla cifra binaria 100).*

Il trasferimento di un'informazione codificata in binario, consente di diminuire il numero dei collegamenti. Così, per indirizzare una memorie di 256 posizioni, occorrerebbero 256 fili per selezionarne una tra 256, cosa difficilmente realizzabile in pratica. La soluzione consiste nel codificare l'indirizzo su 8 bit (bastano in questo modo 8 fili di collegamento) e disporre all'interno della memoria un decodificatore che può selezionare ognuno dei 256 indirizzi delle posizioni di memoria.

CONCETTO DI REGISTRO

Un microprocessore opera su "parole", cioè gruppi di 4, 8, 12 o 16 bit. Per operare su queste parole, tra il microprocessore e la memoria, da una parte, e il microprocessore e le interfaccie, dall'altra, occorrono degli elementi di memoria, temporanea o definitiva, chiamati registri.

Un registro è costituito da un gruppo di "celle" di memoria contenenti un bit ognuna. Un registro di 8 bit, per esempio, contiene 8 celle di memoria e consente la memorizzazione di 8 bit (figura 3). Una posizione di memoria è, di fatto, un registro.

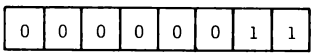


Figura 3 - *Registro a 8 bit contenente il numero 3.*

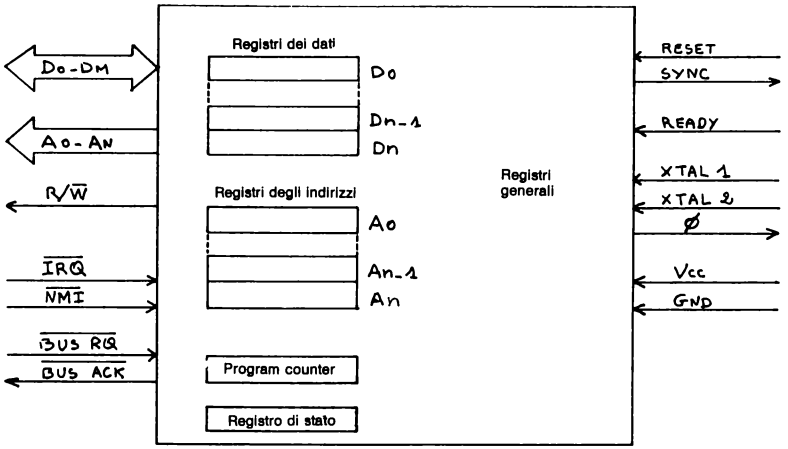
I BUS E I SEGNALI DI COMANDO DI UN MICROPROCESSORE

Il microprocessore comunica con le periferiche tramite tre bus (figure 4):
— il bus degli indirizzi;
— il bus dei dati;
— il bus di controllo.

Il bus degli indirizzi è un insieme di linee unidirezionali, che consentono al microprocessore di selezionare una posizione di memoria o un registro di un'interfaccia. Il microprocessore invia su queste linee, verso una periferica, un indirizzo codificato in binario. La periferica, ricevuto e decodificato l'indirizzo, seleziona il registro corrispondente.

Il numero di linee del bus degli indirizzi determina quella che si chiama la potenza di indirizzamento del microprocessore: per esempio, 16 linee consentono di indirizzare 2^{16} (65236) posizioni di memoria. Notiamo, per inciso, che il concetto di indirizzo è molto vicino a quello che si usa nel linguaggio corrente: la localizzazione di una persona in una città avviene in effetti tramite un indirizzo che contiene la strada e il numero della strada.

Il bus dei dati è costituito da un gruppo di linee bidirezionali sulle quali avvengono gli scambi di dati tra il microprocessore e le periferiche (memoria e interfaccie).



Nome dei piedini	Significato
D_0 — D_M (1)	Bus dei dati (D = data)
A_0 — A_N (2)	Bus degli indirizzi (A = address)
R/$\overline{W}$	Selezione di un'operazione di lettura o scrittura (R = read; W = write)
$\overline{IRQ}$	Richiesta di interruzione mascherabile (IRQ = Interrupt Request)
$\overline{NMI}$	Richiesta di interruzione non mascherabile (NMI = Non Maskable Interrupt)
$\overline{BUS\ RQ}$	Richiesta di accesso diretto alla memoria (BUSRQ = Bus Request)
$\overline{BUS\ ACK}$	Riconoscimento di una richiesta di accesso diretto alla memoria (BUSACK = Bus Acknowledge)
$\overline{RESET}$	Ritorno allo stato iniziale
SYNC	Inizio di un'istruzione
READY	Sincronizzazione con le periferiche
XTAL1, XTAL2	Piedini di collegamento ad un cristallo al quarzo
0	Segnale di clock per le interfaccie
V_{CC}	Piedini di alimentazione
GND	Terra

(1) M = 3, 7, 11 o 15 a seconda che il microprocessore operi su parole di 4, 8, 12 o 16 bit.
(2) N caratterizza la potenza di indirizzamento. Ad esempio, se N = 16, il microprocessore può indirizzare 2^{16} (65536) posizioni di memoria.

Figura 4 - *Organizzazione esterna e interna di un microprocessore.*

Il numero di linee di questo bus dipende dalla lunghezza della parola del microprocessore. A seconda che esso operi su parole di 4, 8, 12 o 16 bit, il bus dati dispone di 4, 8, 12 o 16 linee. Precisiamo, a qusto proposito, che la lunghezza della parola è una caratteristica fondamentale di un microprocessore, poichè determina, in parte, il campo di applicazioni che esso può coprire. Così i microprocessori a 4 bit si prestano alle applicazioni di calcolo (una cifra decimale si rappresenta, in binario, in 4 bit); i microprocessori a 8 bit sono più orientati verso applicazioni di tipo informatico, di telecomunicazioni e di controllo industriale; i microprocessori a 16 bit dovrebbero inserirsi nei campi finora coperti dai minicalcolatori.

Il bus di controllo consiste di un certo numero di segnali di vario genere che assicurano la sincronizzazione tra il microprocessore e le periferiche.

La figura 4 elenca le funzioni più comuni, con sigle (R/$\overline{W}$, $\overline{IRQ}$, $\overline{NMI}$, etc.) eventualmente segnate con una barra orizzontale. Si tratta di sigle inglesi che ricordano la funzione delle linee corrispondenti, per esempio R per "read" (leggere) o W per "write" (scrivere). La presenza o l'assenza della barra orizzontale rappresenta il livello basso o alto (cioè lo stato binario 0 o 1) del segnale di comando. Questi concetti diventeranno più chiari nelle descrizioni dei diversi segnali di comando.

R/$\overline{W}$ (*Read/Write*): selezione di un'operazione di lettura o scrittura. Se il segnale di comando è a livello alto (in genere 5 volt), il microprocessore deve eseguire un'operazione di lettura di una parola di memoria o di un registro; altrimenti (la barra orizzontale indica il livello basso, in genere 0 Volt), deve eseguire un'operazione di scrittura. Il segnale R/$\overline{W}$ è naturalmente associato a un indirizzo presente sul bus A_0 - A_N.

$\overline{IRQ}$ (*Interrupt request*): interruzione mascherabile dal microprocessore. Un segnale di livello basso su questo piedino segnala al microprocessore che una periferica vuole comunicare con lui. Supponiamo che il microprocessore sia collegato ad una tastiera tramite un'interfaccia parallela. Il piedino $\overline{IRQ}$ è a livello alto (ad esempio 5 volt). Quando l'operatore schiaccia un tasto della tastiera, l'interfaccia genera una richiesta di interruzione del microprocessore, portando a livello basso (0 volt) il piedino. Avvisato della richiesta, il microprocessore termina l'esecuzione dell'istruzione in corso, e "salta" a quella parte del programma che gli consente di leggere il carattere emesso dalla tastiera. L'interruzione $\overline{IRQ}$ è mascherabile, cioè può essere disabilitata da programma. Torniamo all'esempio della tastiera. Se $\overline{IRQ}$ è mascherata la pressione su un tasto della tastiera non ha effetto.

$\overline{NMI}$ (*Non-maskable interrupt*): interruzione non mascherabile dal microprocessore. Questo piedino ha la stessa funzione di $\overline{IRQ}$, con l'unica differenza che questa interruzione non è mascherabile. È di fondamentale importanza in quanto consente, in caso di caduta dell'alimentazione, di salvare i contenuti dei registri del microprocessore in una memoria alimentata da una batteria, e di recuperarli quando torna l'alimentazione, in modo da continuare l'esecuzione del programma come se non fosse successo niente. Per spiegare la funzione di $\overline{NMI}$, supponiamo

che il microprocessore sia inserito in un programmatore elettronico di una lavatrice. Se c'è una caduta di tensione, il programmatore, a differenza di un analogo dispositivo meccanico, perde tutte le informazioni, per cui se la nostra macchina era nella fase del lavaggio, rischia di ripartire, quando torna la tensione, da una fase diversa. L'interruzione non mascherabile salva, in caso di caduta di tensione, le informazioni fondamentali, assicurando così una ripartenza corretta (nel nostro esempio, dalla fase di lavaggio).

$\overline{\text{BUSRQ}}$ (*Bus Request*): richiesta di accesso al bus. Applicando un segnale di livello basso al piedino $\overline{\text{BUSRQ}}$, una periferica veloce (come un disco), un altro microprocessore o un elaboratore provoca l'arresto del microprocessore, che, dopo aver terminato l'esecuzione dell'instruzione in corso, si scollega dai bus degli indirizzi e dei dati, e da alcune linee del bus di comando. Ovviamente non si scollega nel senso proprio del termine, ma diventa di fatto insensibile allo stato delle linee del bus.

$\overline{\text{BUSACK}}$ (*Bus acknowledge*): riconoscimento di una richiesta di accesso al bus. È la risposta alla richiesta precedente. Applicando un segnale di livello basso su questo piedino, il microprocessore segnala al sistema che ha emesso $\overline{\text{BUSRQ}}$ che può disporre dei bus. Questi comandi consentono trasferimenti rapidi tra una periferica veloce e la memoria del microprocessore (accesso diretto alla memoria), possono far funzionare diversi microprocessori collegati agli stessi bus (sistemi a multi-processori) e rendono possibile il dialogo tra un elaboratore centrale e dei terminali.

RESET: ritorno allo stato inziale. Un segnale di livello basso su questo piedino determina l'inizializzazione del microprocessore.

SYNC: inizio di un'istruzione

READY: sincronizzazione con le memorie o periferiche lente. Fin quando una memoria o una periferica non sono pronte a rispondere ad una chiamata del microprocessore, il piedino READY viene mantenuto ad un livello basso. Il microprocessore resta in attesa che la memoria o la periferica in questione siano in grado di dare una risposta, inviando un segnale di livello alto sullo stesso piedino.

XTAL 1, XTAL 2: piedini di collegamento ad un cristallo al quarzo. Il microprocessore è temporizzato da un segnale di clock, emesso da un cristallo al quarzo, collegato tra i piedini XTAL 1, e XTAL 2. Questi segnali servono di riferimento temporale per il microprocessore.

0: segnale di clock per le periferiche. Questo segnale viene elaborato dal microprocessore, a partire da quello emesso dal cristallo al quarzo, e inviato verso le periferiche, come riferimento temporale per i dispositivi che ne hanno bisogno.

V_{CC} e GND: piedini di alimentazione e di terra. I microprocessori sono alimentati oggi sempre più spesso da una sola sorgente di tensione (+5 volt), mentre i vecchi modelli ne avevano diverse.

ORGANIZZAZIONE DI UN MICROPROCESSORE

Un microprocessore contiene parecchie migliaia di transistors, e non è quindi il caso di descriverlo nei minimi particolari. D'altra parte non sarebbe neanche molto utile per il lettore.

Un microprocessore è costituito da tre parti principali:

— unità di controllo;
— unità aritmetico-logica;
— i registri.

L'unità di controllo decodifica le istruzioni prelevate dalla memoria di programma ed elabora i segnali di comando necessari all'esecuzione di un'istruzione.

L'unità aritmetico-logica esegue le operazioni aritmetiche e logiche.

I registri sono di due tipi: alcuni sono accessibili dal programma dell'utente, altri no. Solo i primi interessano il programmatore, e si dividono in tre categorie (figura 4):

— i registri dei dati (D_0 — D_N)
— i registri degli indirizzi (A_0 — A_N);
— il contatore di programma (Program Counter) e il registro di stato.

I registri dei dati consentono la memorizzazione temporanea delle informazioni negli spostamenti tra l'unità aritmetico-logica, le memorie e le interfaccie. La loro lunghezza è uguale a quella della parola del microprocessore: 8 bit se il microprocessore opera su 8 bit.

I registri di indirizzo, detti anche puntatori, contengono indirizzi delle posizioni di memoria, che vengono inviati al bus di indirizzo con un'istruzione particolare che consente l'accesso a tali posizioni. La lunghezza di un registro di indirizzo è uguale a quella del bus di indirizzo.

Il fatto di avere diversi registri di indirizzo consente di "puntare" diverse zone di memoria durante l'esecuzione di un programma.

Supponiamo per esempio che il microprocessore debba cercare in un insieme di elementi contenuti in un blocco di memoria un gruppo di questi caratterizzati da un indice comune. Il principio della ricerca è descritto nella figura 5.

Ogni elemento è descritto in quattro byte (quattro parole di 8 bit), e il primo di questi contiene l'indice, che potrebbe essere l'età di un individuo, la sua professione oppure la città dove risiede, se il blocco di memoria contiene un elenco di persone. In base all'indice scelto dall'utente, ad esempio l'età, il microprocessore seleziona tutti gli elementi del blocco di memoria che hanno indice uguale a quello scelto. Se l'indice è 30 anni, il microprocessore cerca all'interno del blocco tutti gli individui di 30 anni, operando nel modo seguente: legge l'indice del primo articolo, lo confronta

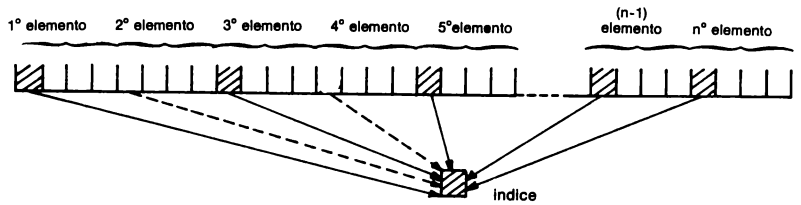


Figura 5 - *Ricerca all'interno di un blocco di memoria degli elementi caratterizzanti da un dato indice.*

con quello scelto. Se sono uguali (vedi figura 5) il microprocessore copia l'elemento in una zona determinata della memoria. Altrimenti legge l'indice dell'elemento seguente. Queste operazioni vengono ripetute su tutti gli elementi, fino a che è stato analizzato tutto il blocco. per questa ricerca, il microprocessore ha bisogno di due puntatori: uno che esplora passo passo il blocco di memoria, l'altro che punta verso la zona in cui vengono ricopiati gli elementi selezionati (zona di ricopiatura nella figura 6). L'indice scelto dall'utente è contenuto in un registro dei dati; gli indici degli elementi sono trasferiti uno dopo l'altro dalla memoria verso un altro registro dei dati, e il microprocessore, dopo ogni trasferimento, esegue il confronto dei contenuti di questi registri. Quando sono uguali invia l'elemento verso la zona di ricopiatura.

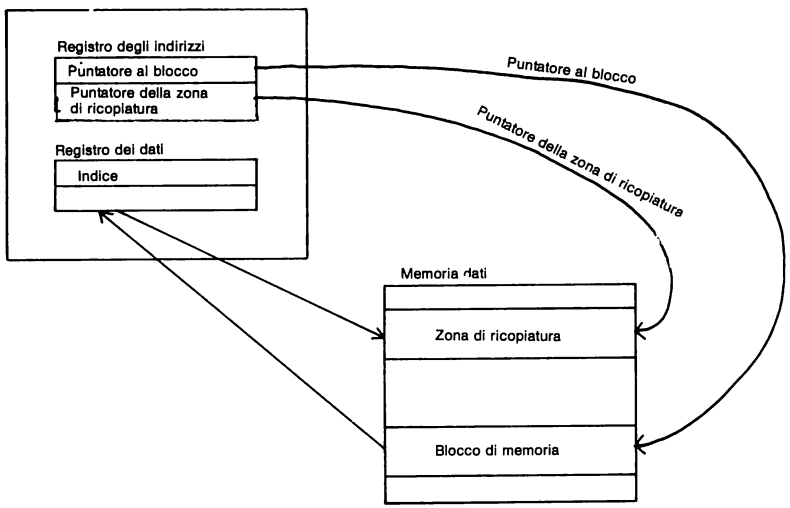


Figura 6 - *Registri utilizzati in un'operazione di ricerca in memoria.*

Il puntatore dello stack è un registro di indirizzo particolare che punta una particolare zona della memoria, che si chiama "area di stack". Viene automaticamente decrementato dopo ogni traferimento di una parola in questa zona, e incrementato dopo ogni prelievo. Il puntatore dell'area di stack ha una funzione particolare nella richiesta di interruzione e nelle chiamate ai sottoprogrammi. Ci torneremo in seguito.

Il Program Counter (o contatore delle istruzioni) presiede all'esecuzione di un programma. Viene caricato, inizialmente con l'indirizzo della prima istruzione del programma, e in seguito segnala al microprocessore gli indirizzi delle istruzioni che devono essere seguite. Mentre il microprocessore legge nella memoria di programma una istruzione, la decodifica e l'esegue, il contatore prosegue fino all'indirizzo dell'istruzione successiva e via di seguito.

Il microprocessore in genere esegue le istruzioni di un programma in modo sequenziale, cioè una dopo l'altra. Nel caso però di alcune istruzioni (salti, chiamate ai sottoprogrammi, richiesta di interruzione) l'esecuzione sequenziale del programma viene modificata: nel Program Counter viene caricato l'indirizzo di salto, e l'indirizzo di ritorno al programma principale viene conservato per essere riutilizzato in seguito.

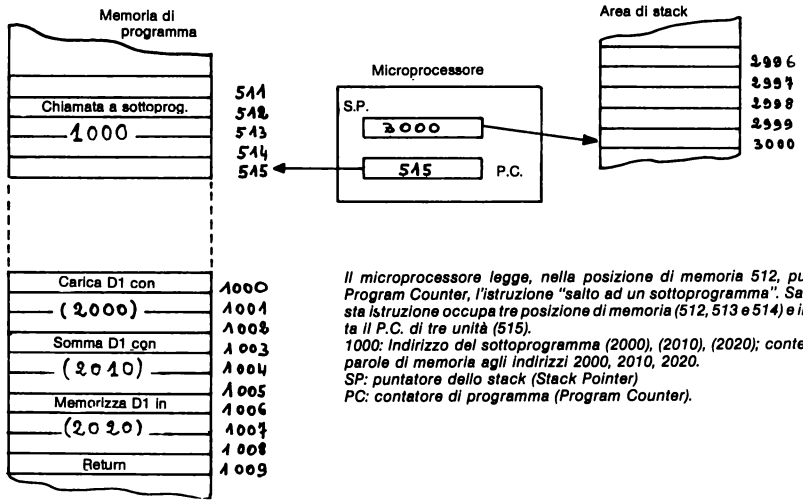
Il registro di stato, ultimo da esaminare nella figura 4, contiene un certo numero di bit, a 0 o a 1, a seconda che il risultato ottenuto dopo l'esecuzione di alcune istruzioni sia positivo, negativo, nullo, con riporto o senza, etc. Lo svolgimento del programma potrà essere modificato a seconda di questi risultati.

Vedremo in seguito che esistono istruzioni di salto incondizionato ad un indirizzo dato, e altre di salto condizionato, che causano il passaggio ad un'altra parte del programma a seconda del risultato di un'operazione (per esempio, salta se e soltanto se il rgistro di stato indica che il risultato di un'operazione è nullo). In genere i salti (incondizionati o condizionati) servono per uscire dal programma principale e passare ad un sottoprogramma. Durante l'esecuzione di quest'ultimo, quella del programma principale è interrotta. Al momento del salto al sottoprogramma, il suo indirizzo iniziale viene caricato nel Program Counter, dopodichè viene eseguito sequenzialmente come se fosse il programma principale. Quando l'esecuzione del sottoprogramma è terminata nel Program Counter viene caricato l'indirizzo di ritorno al programma principale (che, come vi ricorderete, era stato conservato).

La figura 7 descrive, in quattro fasi, lo svolgimento di queste operazioni. Il programma principale è memorizzato a partire dall'indirizzo 500. Nella posizione 512 c'è un'istruzione di salto ad un sottoprogramma, il cui indirizzo iniziale è 1000. Poichè l'istruzione di salto occupa tre posizioni di memoria, l'indirizzo di ritorno al programma principale sarà 515 (512 + 3). Il microprocessore trasferisce il numero 515 nell'area di stack, agli indirizzi 3000 e 2999, poichè 515 occupa due posizioni di memoria. Dopodichè nel Program Counter viene caricato l'indirizzo di inizio del sottoprogramma (1000). Questo viene eseguito fino all'istruzione 1009 (ritorno al

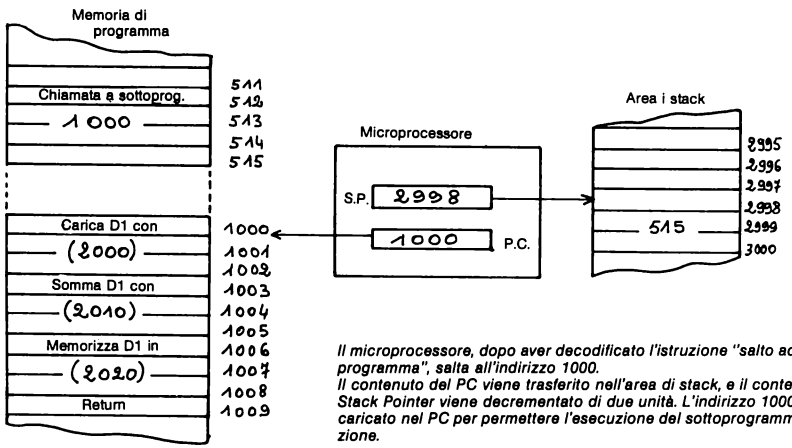
programma principale). Nel Program Counter viene allora caricato il contenuto dell'area di stack, nel nostro caso 515. Notate nella figura 7 il funzionamento particolare del puntatore dello stack.

Le uscite dal programma principale non avvengono solo in corrispondenza di istruzioni di salto o di chiamata a un sottoprogramma. Possono avvenire anche come conseguenza di eventi esterni o interni al microprocessore. Ad esempio l'arrivo di un'istruzione errata è un evento interno che può provocare un'interru-



Il microprocessore legge, nella posizione di memoria 512, puntata dal Program Counter, l'istruzione "salto ad un sottoprogramma". Sa che questa istruzione occupa tre posizione di memoria (512, 513 e 514) e incrementa il P.C. di tre unità (515).
1000: Indirizzo del sottoprogramma (2000), (2010), (2020); contenuti delle parole di memoria agli indirizzi 2000, 2010, 2020.
SP: puntatore dello stack (Stack Pointer)
PC: contatore di programma (Program Counter).

Figura 7a - Lettura dell'istruzione di chiamata.



Il microprocessore, dopo aver decodificato l'istruzione "salto ad un sottoprogramma", salta all'indirizzo 1000.
Il contenuto del PC viene trasferito nell'area di stack, e il contenuto dello Stack Pointer viene decrementato di due unità. L'indirizzo 1000 viene poi caricato nel PC per permettere l'esecuzione del sottoprogramma di addizione.

Figura 7b - Salto al sottoprogramma.

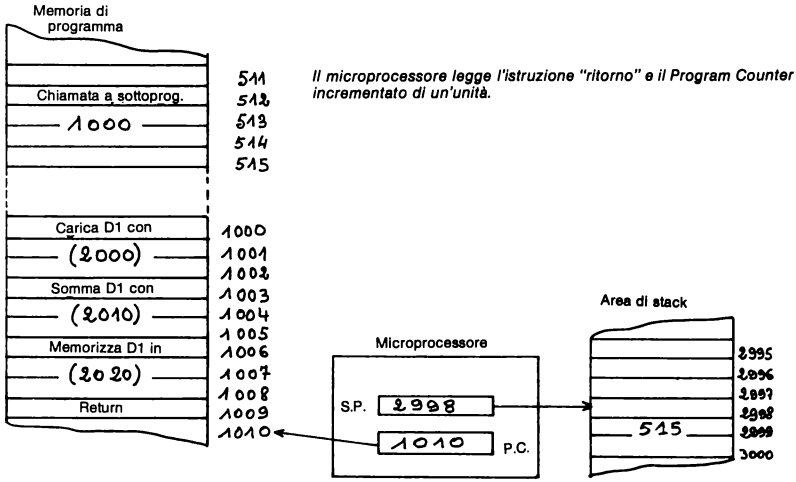


Figura 7c - Fine dell'esecuzione del sottoprogramma.

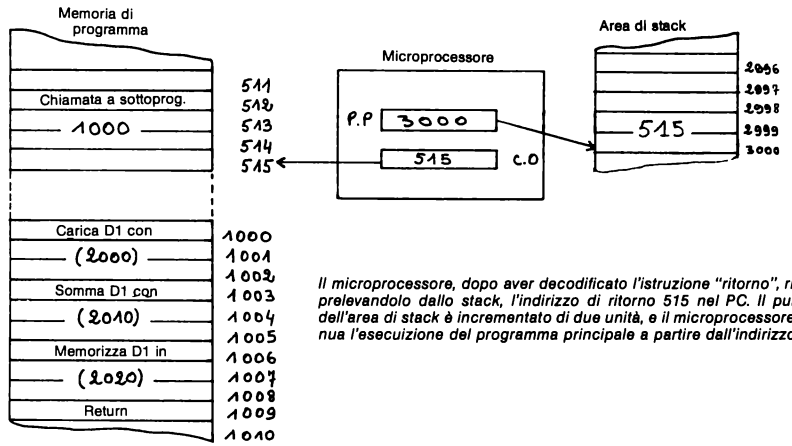


Figura 7d - Ritorno al programma principale.

Figura 7 - *Procedura di chiamata ad un sottoprogramma e ritorno al programma principale.*

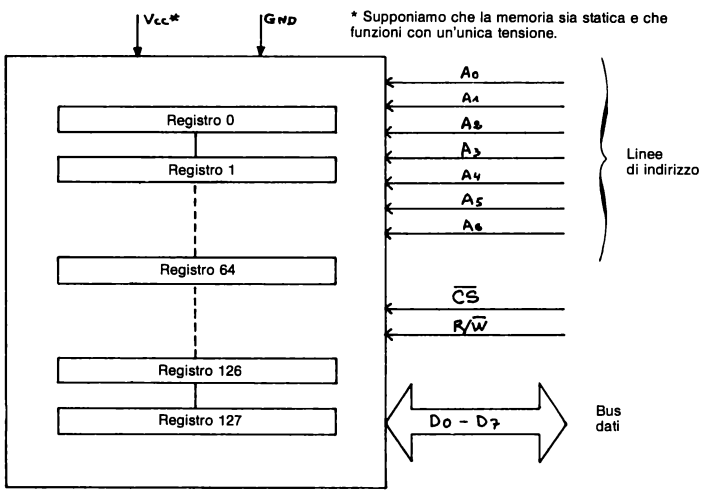
zione del programma principale e un salto ad un programma di servizio, che segnala il fatto all'utente. Analogamente, quando una periferica è pronta ad inviare un carattere, invia un segnale di richiesta di interruzione sul piedino $\overline{IRQ}$. Se la richiesta è accolta, l'esecuzione del programma principale viene interrotta dopo

l'esecuzione dell'istruzione in corso, e il microprocessore salta ad un sottoprogramma di servizio dell'interruzione che gestisce l'ingresso di un carattere dalla periferica in questione.

Nel caso di un'interruzione casuale, dovuta a un evento non programmato il microprocessore non si limita a conservare nell'area di stack l'indirizzo di ritorno, ma salva anche il contenuto del registro di stato ed eventualmente il contenuto dei registri dei dati e degli indirizzi. Una volta eseguito il sottoprogramma di servizio dell'interruzione, il microprocessore ricarica nei registri, prelevandoli dall'area di stack, i loro contenuti originali, e torna al programma principale. Ricordiamo che l'area di stack è una zona della memoria dati, organizzata in modo che l'ultimo dato entrato è l'ultimo a uscire (LIFO = last in, first out).

ORGANIZZAZIONE DI UNA MEMORIA RAM

La figura 8 dà un esempio di una memoria RAM (Random Access Memory), che contiene 128 registri a 8 bit nei quali si possono leggere o scrivere informazioni in qualsiasi momento. Per accedere ad una posizione di memoria bisogna disporre di un indirizzo codificato su 7 bit (A_0 — A_6) che, decodificato all'interno del circuito, consente di selezionare una posizione su 128. Associato all'indirizzo, il segnale $\overline{CS}$ abilita un solo dispositivo tra molti, e il segnale R/$\overline{W}$ consente di distinguere tra un'operazione di lettura e una di scrittura.



Nome dei piedini	Significato
A_0 — A_6	Linee degli indirizzi
$\overline{CS}$	Selezione dispositivo (Chip Select)
$R/\overline{W}$	Selezione lettura/scrittura
D_0 — D_7	Bus dei dati
V_{CC} — GND	Alimentazione e massa

Figura 8 - *Organizzazione esterna e interna di una memoria RAM di 128 parole di 8 bit.*

Il dato che viene scambiato tra il microprocessore e la memoria passa sul bit dei dati (D_0 — D_7, nel caso di un microprocessore a 8 bit).

Riassumendo, una memoria comprende:

— una o più linee di selezione dispositivo: $\overline{CS}$ (chip select = selezione dispositivo);
— delle linee di indirizzo che consentono di selezionare, all'interno del dispositivo, una posizione tra n;
— una linea di selezione lettura/scrittura;
— delle linee dati (bus dei dati).

Il numero delle linee di indirizzo dipende evidentemente dal numero delle posizioni di memoria contenute nel dispositivo; il numero delle linee dati è una caratteristica della memoria. Nel caso di un microprocessore a 8 bit, che utilizza memorie di 256 parole di 4 bit ciascuna (cioè 1024 bit), la realizzazione di una memoria di 256 parole di 8 bit avverrà associando due chip di memoria (figura 9).

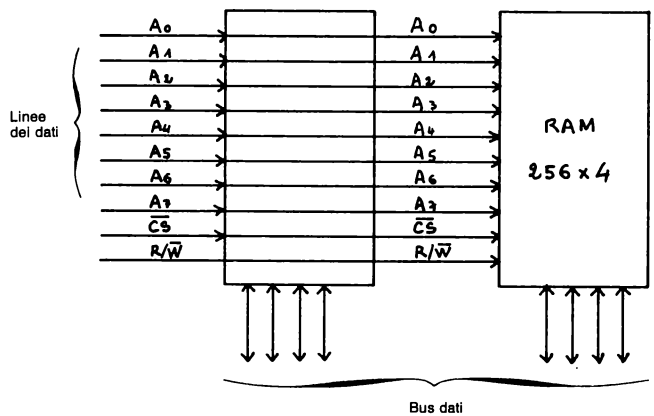
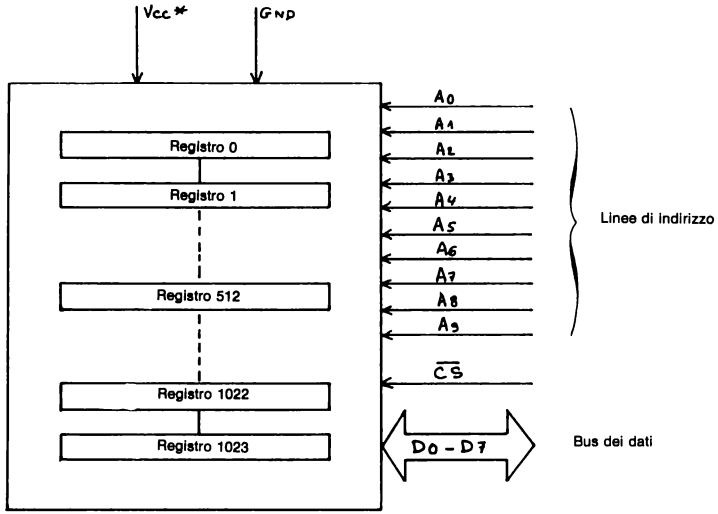


Figura 9 - *Memoria RAM di 256 parole di 8 bit, realizzata con chip 256 x 4.*



** Supponiamo che la memoria sia statica e funzioni con una sola tensione*

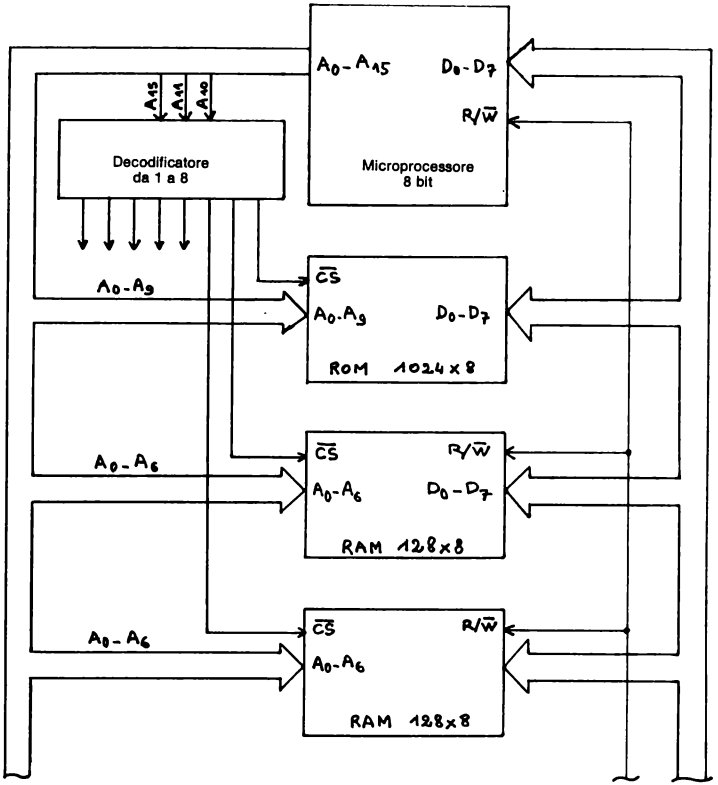
Nome dei piedini	Significato
A_0 — A_9	Linee degli indirizzi
$\overline{CS}$	Selezione dispositivo (Chip Select)
D_0 — D_7	Bus dei dati
V_{CC} — GND	Alimentazione e massa

Figura 10 - *Organizzazione esterna e interna di una ROM di 1024 parole di 8 bit.*

ORGANIZZAZIONE DI UNA MEMORIA ROM

La figura 10 è un esempio di memoria ROM di 1024 parole di 8 bit, la cui organizzazione è molto simile a quella della RAM della figura 8. La differenza essenziale è che manca la linea R/$\overline{W}$, visto che una ROM si può solo leggere.

Essa comprende 1024 registri a sola lettura, indirizzabili con 10 linee di indirizzo (da A_0 a A_9), una linea di selezione dispositivo $\overline{CS}$, e 8 linee di dati (da D_0 a D_7). La figura 11 mostra un esempio del collegamento ad un microprocessore di una ROM di 1024 parole di 8 bit, con indirizzi tra 0 e 1023, e di due RAM di 128 parole di 8 bit, con indirizzi da 1024 a 1151 l'una, da 2048 a 2175 l'altra. La figura 14 mostra come sono determinati questi indirizzi.



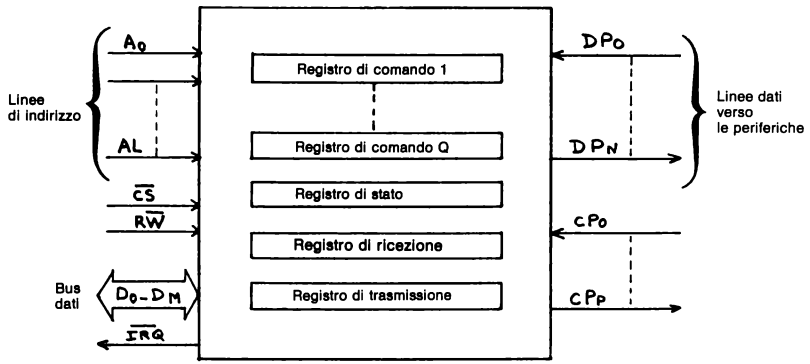
A_{15}*	A_{11}	A_{10}	Stato del decodificatore	
0	0	0	Stato 0 - scelta della ROM	
0	0	1	Stato 1 - scelta di una RAM	
0	1	0	Stato 2 - scelta dell'altra RAM	
0	1	1	Stato 3	
1	0	0	Stato 4 - scelta della 1ª interfaccia	utilizzate nella fig. 13
1	0	1	Stato 5 - scelta della 2ª interfaccia	
1	1	0	Stato 6 - scelta della 3ª interfaccia	
1	1	1	Stato 7	

(La linea A15 in realtà è inutile in questa applicazione, poichè ci basterebbe un decodificatore da 1 a 4. Ciononostante abbiamo scelto un decodificatore da 3 a 8, con ingressi A_{15}, A_{11}, A_{10}, poichè A_{15} ci sarà utile quando collegheremo delle interfaccie. Notiamo che A_{15} resta a 0 quando si seleziona una delle tre memorie).*

Figura 11 - *Collegamento di un microprocessore a 8 bit a memorie ROM e RAM.*

ORGANIZZAZIONE DI UN'INTERFACCIA

Il microprocessore deve poter dialogare con le sue periferiche: tastiera, lettore di nastro perforato, unità a floppy disk, stampanti, sensori, etc. Questo dialogo è reso possibile da interfaccie programmabili, dove "programmabili" significa che il loro ruolo in una certa applicazioni è definito da programma. Quindi un'unica interfaccia può adattarsi ad ambienti e applicazioni diverse.



Nome dei piedini	Significato
A_0 — A_L	Piedini di selezione di un registro interno all'interfaccia. L dipende dal numero di registri accessibili.
$\overline{CS}$	Linea di selezione dispositivo.
$R/\overline{W}$	Linea di selezione del modo ingresso/uscita.
D_0 — D_M	Bus dei dati —M dipende dalla lunghezza della parola su cui opera il microprocessore.
$\overline{IRQ}$	Richiesta di interruzione per il microprocessore
DP_0 — DP_N	Linee dei dati verso le periferiche. N dipende dal tipo di interfaccia.
CP_0 — CP_P	Linee di comando e di stato, per e dalle periferiche. p dipende dal tipo di interfaccia.

Figura 12 - *Organizzazione esterna e interna di un'interfaccia.*

D'altra parte non è possibile realizzare interfaccie universali in un chip di 24 o 40 piedini. Ogni tipo di interfaccia copre un campo di applicazioni ben preciso, all'interno del quale presenta una flessibilità sufficiente per adattarsi a diversi tipi di applicazioni. Nel capitolo seguente studieremo i diversi tipi di interfaccie.

La figura 12 descrive lo schema tipico di un'interfaccia, che comprende in genere quattro tipi di registri accessibili dall'utente:

— *i registri di comando e di controllo* che definiscono la funzione dell'interfaccia e sono a sola scrittura. Quando ce n'è più di uno, esiste di solito un registro primario, accessibile direttamente con le abituali tecniche di indirizzamento, e dei registri secondari accessibili indirettamente tramite quello primario: l'indirizzamento di un registro secondario implica quindi l'indirizzamento preliminare del registro primario, nel quale viene scritto, il numero del registro secondario che si vuole poi utilizzare.
— *il registro di stato* che contiene un certo numero di bit, posizionati a 0 o a 1 a seconda dello stato delle comunicazioni con le periferiche (richiesta di interruzione da parte di una periferica che vuole dialogare con il microprocessore, errore nella trasmissione, errore nella ricezione, etc.). Il registro di stato è a sola lettura.
— *il registro di ricezione* che riceve dati che provengono dalle periferiche, ed è a sola lettura.
— *il registro di trasmissione* che contiene i dati provenienti dal microprocessore e diretti verso le periferiche. Questo registro è a sola scrittura.

Le linee che collegano un microprocessore ad un'interfaccia sono:

— la linea di selezione dispositivo ($\overline{CS}$);
— le linee di selezione dei registri dell'interfaccia, che provengono di solito dal bus degli indirizzi del microprocessore (A_0 — A_L);
— la linea di selezione del modo ingresso/uscita di un dato ($R/\overline{W}$);
— il bus dei dati (D_0 — D_N);
— la linea di richiesta di interruzione $\overline{IRQ}$.

Le linee che collegano l'interfaccia alle periferiche sono:

— linee di dati (DP_0 — DP_N), che trasferiscono i dati tra l'interfaccia e le periferiche;
— linee di comando e di stato, per e dalle periferiche (CP_0 — CP_P) che servono o per segnalare alle periferiche che il microprocessore sta per trasmettere un dato, o che una periferica desidera comunicare con il microprocessore. Queste linee assicurano anche il controllo della trasmissione e della ricezione da parte delle periferiche.

INDIRIZZAMENTO DELLE INTERFACCE E DEI CIRCUITI DI MEMORIA

In figura 13 è descritto un esempio di collegamento tra circuiti di memoria e interfaccie, e un microprocessore a 8 bit. La selezione tra i circuiti di memoria e le interfaccie si fa con la linea A_{15}. Quando $A_{15} = 0$ (livello basso), il microprocessore esegue un'operazione in memoria. Quando $A_{15} = 0$ (livello basso), il microprocessore esegue un'operazione in memoria. Quando $A_{15} = 1$ (livello alto), il microprocessore opera sui dispositivi di ingresso/uscita. La selezione di uno dei tre chip di

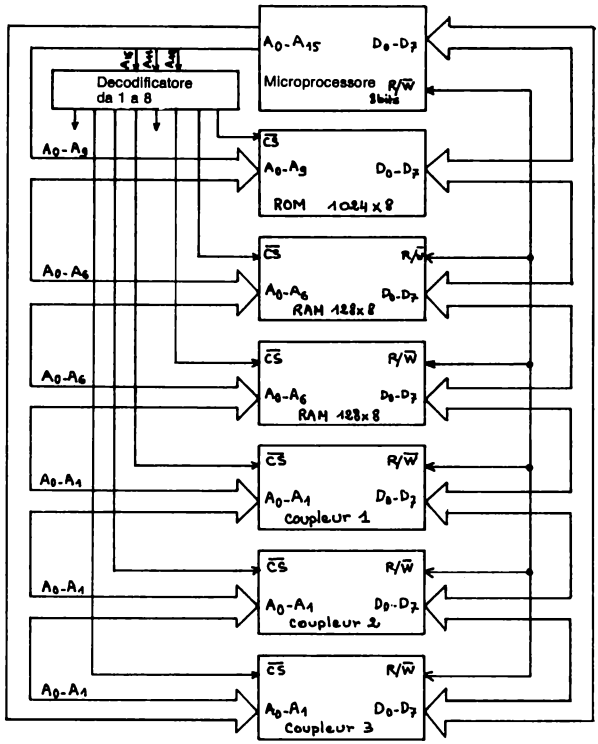


Figura 13 - *Collegamento di un microprocessore a circuiti di memoria e di interfaccia.*

32768	16384	8192	4096	2048	1024	512	256	128	64	32	16	8	4	2	1		
A_{15}	A_{14}	A_{13}	A_{12}	A_{11}	A_{10}	A_9	A_8	A_7	A_6	A_5	A_4	A_3	A_2	A_1	A_0	Decimale	Memorie o Interfaccie
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ROM
0	0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1023	
0	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	1024	RAM 1
0	0	0	0	0	1	0	0	0	1	1	1	1	1	1	1	1151	
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	2048	RAM 2
0	0	0	0	1	0	0	0	0	1	1	1	1	1	1	1	2175	
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	32768	Coupleur 1
1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	32771	
1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	33792	Coupleur 2
1	0	0	0	0	1	0	0	0	0	0	0	0	0	1	1	33795	
1	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	34816	Coupleur 3
1	0	0	0	1	0	0	0	0	0	0	0	0	0	1	1	34819	

Le linee di indirizzo che intervengono nella determinazione degli indirizzi di un circuito di memoria o di un'interfaccia, nelle figure 11 e 13, sono quelle da A_0 a A_9, A_{10}, A_{11} e A_{15}. Le altre non intervengono, e noi le supponiamo a zero.
Il valore dell'indirizzo più alto corrispondente ad un chip si ottiene associando ad ogni 1 di una fila il valore decimale corrispondente di quella colonna (scritto sopra al simbolo della linea), e sommando i diversi valori decimali trovati. Per esempio, l'indirizzo più alto della ROM è:
512 + 256 + 128 + 64 + 32 + 16 + 8 + 4 + 2 + 1 = 1023

Figura 14 - *Determinazione degli indirizzi dei circuiti di memoria e delle interfaccie.*

memoria si fa con le linee A_{10} e A_{11}; lo stesso per le interfaccie. Il decodificatore da 1 a 8 riceve le tre linee di indirizzo A_{15}, A_{11} e A_{10} e invia in uscita 8 segnali, sei dei quali servono a selezionare un chip tra i sei delle memorie e delle interfaccie (vedi figura 11). Gli indirizzi dei diversi componenti sono dati nella figura 14.

INIZIALIZZAZIONE DI UN'INTERFACCIA

Un micro-elaboratore deve poter comunicare con le periferiche, pur garantendo l'esecuzione del programma principale. Ogni periferica è dunque collegata al microprocessore tramite un'interfaccia, e quando vuole comunicare invia una richiesta di dialogo sulla linea di stato STB (Strobe), che fa parte delle linee di comando e di stato dell'interfaccia associata. Questa richiesta mette a 1 il bit I del registro di stato (figura 15).

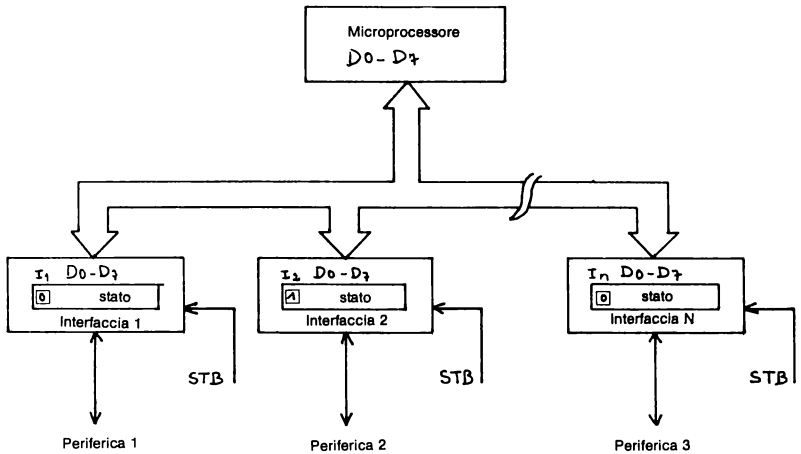


Figura 15 - *Gestione degli ingressi/uscite interamente tramite software.*

Il microprocessore può gestire in diversi modi il dialogo con le periferiche. In istanti di tempo ben precisi, definiti dal programmatore, legge il contenuto del registro di stato dell'interfaccia 1, e controlla se il bit I_1 è a 1 (figura 16). In caso affermativo, salta al sottoprogramma di ingresso/uscita associato alla periferica 1. In caso negativo legge il contenuto del registro di stato dell'interfaccia 2, ed esegue il test sul bit I_2, e così di seguito, fino a quando trova una periferica pronta a comunicare, oppure fino a quando ha controllato tutti i registri di stato delle interfaccie. In quest'ultimo caso ritorna al programma principale. Questo metodo, che consiste nel controllare periodicamente tutti i registri di stato, anche quando non c'è nessuna richiesta di dialogo, ha tempi lunghi e riduce l'efficienza del sistema. È utilizzato nelle applicazioni in cui il microprocessore non è sovraccarico di lavoro.

In altri casi, si usa il metodo delle interruzioni, che consiste nell'interrompere il microprocessore ogni volta che una periferica è pronta a comunicare. In questo modo di funzionamento, la linea di interruzione $\overline{IRQ}$ di ogni interfaccia è collegata al piedino di interruzione $\overline{IRQ}$ del microprocessore (figura 17), che è tenuto a 5 volt tramite una resistenza. Se nessuna periferica interrompe il microprocessore, le linee $\overline{IRQ}$ di tutte le periferiche sono allo stato alto, e così anche il piedino $\overline{IRQ}$ del microprocessore. Se una o più periferiche vogliono dialogare con il microprocessore, una o più linee $\overline{IRQ}$ delle interfaccie si trovano allo stato basso, provocando sul piedino $\overline{IRQ}$ del microprocessore una richiesta di interruzione. Il microprocessore finisce l'istruzione in corso, e accetta questa interruzione, se è abilitato a farlo (la

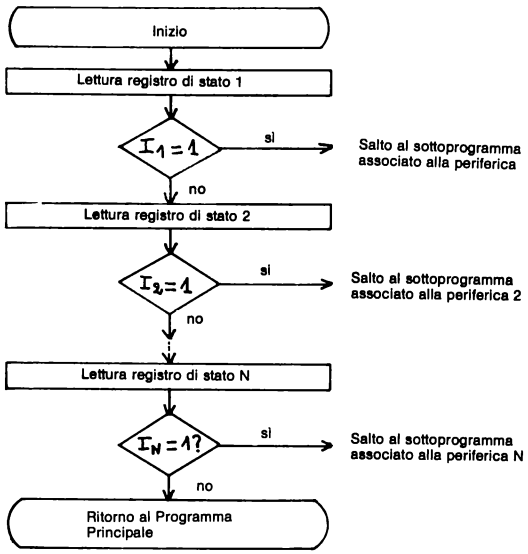


Figura 16 - *Determinazione tramite software della periferica pronta a comunicare.*

abilitazione e la disabilitazione delle interruzioni si fa da programma, posizionando a 1 o a 0 il bit I del registro di stato del microprocessore). Il riconoscimento di un'interruzione da parte del microprocessore provoca il mascheramento di tutte le altre richieste per tutta la durata del servizio. Questo comincia, come nel caso precedente, con il test dei bit I dei registri di stato (figura 16) e con l'esecuzione delle sequenze di ingresso/uscita associate alla periferica che ha generato l'interruzione.

Il metodo delle interruzioni presenta il vantaggio di non disturbare l'esecuzione del programma principale, se non quando una periferica vuole comunicare con il microprocessore. L'efficienza del sistema migliora sensibilmente. È necessaria comunque una fase di indagine (polling) per determinare la periferica che ha generato la interruzione, e per questo il tempo di risposta ad una richiesta di

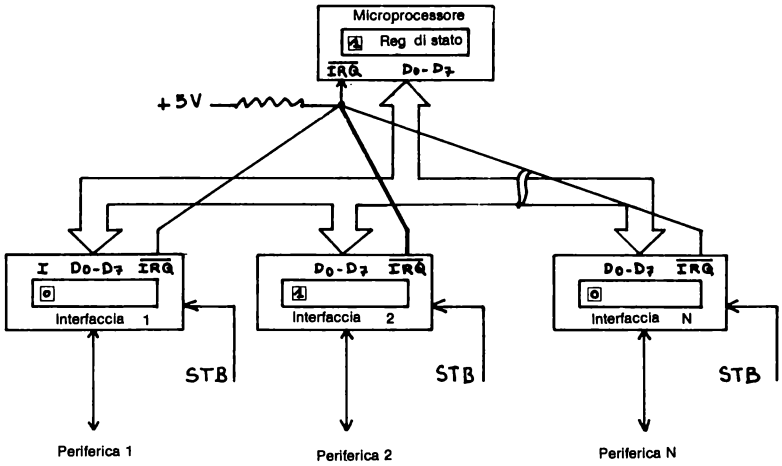


Figura 17 - *Gestione degli ingressi/uscite tramite interruzione e polling.*

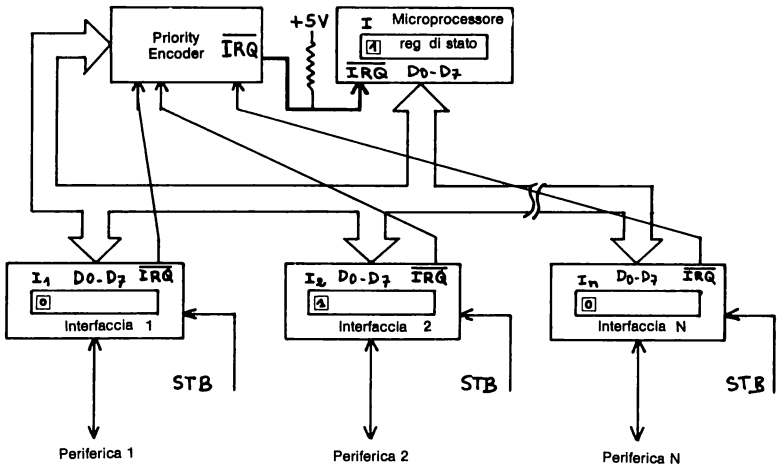


Figura 18 - *Gestione degli ingressi/uscite tramite interruzioni e priority encoder.*

interruzione può essere molto lungo.

Nelle applicazioni, in cui le operazioni di ingresso uscita hanno una grande importanza, conviene eseguire questa indagine tramite hardware (priority encoder), come si vede nella figura 18. Il priority encoder (codificatore di priorità) riceve le richieste di interruzione, e se ce ne è più di una, determina la periferica con la priorità più elevata, controlla che il livello di questa priorità sia superiore ad un valore definito da programma, e genera una richiesta di interruzione alla CPU e un indirizzo di salto ad un sottoprogramma di ingresso/uscita associato alla periferica in questione.

CAPITOLO 3

I DIVERSI TIPI DI INTERFACCIA

Il microprocessore comunica con le periferiche tramite interfaccie "programmabili", la cui funzione, cioè, è definita da programma. Tali interfaccie non hanno quindi una struttura rigida, e possono adattarsi ad ambienti diversi, se programmate in modo opportuno. Nonostante ciò, ogni tipo di interfaccia copre un campo limitato di applicazioni, all'interno del quale è abbastanza flessibile da essere adattato ai vari casi particolari. In questo capitolo affrontiamo lo studio dei diversi tipi di interfaccia.

INTERFACCIA PARALLELA

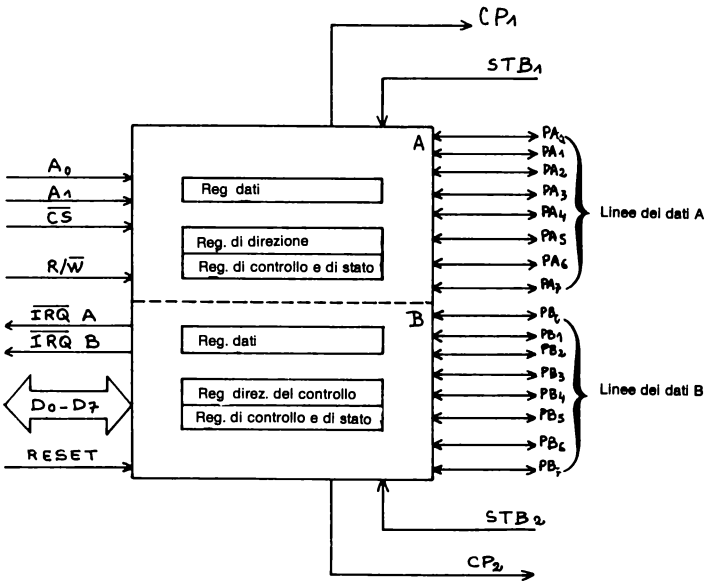
Questo tipo di interfaccia smista i dati provenienti dal microprocessore verso diverse periferiche, e centralizza i dati provenienti dalle periferiche per trasmetterli al microprocessore. È una sorta di centro di scambio dei dati che permette di collegare, sul bus dei dati del microprocessore, due o tre periferiche (che devono operare su parole della stessa lunghezza del microprocessore).

la figura 1 dà l'esempio di un'interfaccia che collega un microprocessore a 8 bit a due periferiche. Questo esempio può essere generalizzato ai microprocessori a 4, a 12 o a 16 bit. Le linee che collegano l'interfaccia alle periferiche sono:

— sedici linee di dati, suddivise in due gruppi di 8; ogni linea può essere programmata individualmente come ingresso o come uscita;
— quattro linee di comando e di stato, che comprendono due linee STB, per segnalare che le periferiche sono pronte a comunicare, e due linee di comando per le periferiche (CP). Con le due linee CP il microprocessore invia alle periferiche comandi, segnali di ricezione dei dati, e segnali di trasmissione dei dati.

L'interfaccia parallela è costituita praticamente da quattro posizioni di memoria, indirizzabili con le due linee di indirizzo A_0 e A_1, che corrispondono ai registri seguenti:

— *registri dei dati*: sono associati ad ogni gruppo di 8 linee, chiamato porta, e servono alla memorizzazione temporanea dei dati che provengono dalle periferiche o dal microprocessore;



Nome dei piedini	Significato
A_0, A_1	Linee di selezione di uno di quattro gruppi di registri
$\overline{CS}$	Linee di selezione dispositivo
$R/\overline{W}$	Linea di selezione del modo di lettura o scrittura
IRQ_A	Richiesta di interruzione della periferica A
$\overline{IRQ}_B$	Richiesta di interruzione dalla periferica B
D_0 - D_7	Bus dei dati
RESET	Ritorno allo stato iniziale
STB_1, STB_2	Linee di controllo dalle periferiche all'interfaccia
CP_1, CP_2	Linee di comando dall'interfaccia alle periferiche
PA_0 — PA_7	Linee di dati della porta A
PB_0 — PB_7	Linee di dati della porta B

Figura 1 - *Organizzazione esterna ed interna di un'interfaccia parallela.*

— *registri di controllo*: definiscono il ruolo dell'interfaccia; ogni registro di controllo è composto di fatto di due registri:

- *il registro di controllo e di stato*, che determina la funzione esatta dei piedini STB e CP e registra lo stato dei piedini STB;
- *il registro di direzione*, che definisce la funzione delle linee dei dati. Ad ognuna di queste è associato un bit nel registro, e a seconda che sia a 0 o a 1 (viene posizionato da programma), la linea corrispondente è un ingresso o un'uscita.

Notiamo che ci sono sei registri ma solo due linee di indirizzo, con le quali si possono indirizzare dunque solo quattro registri; l'accesso a due dei sei registri è quindi indiretto.

Collegamento di un microprocessore ad un plotter

La figura 2 da un esempio di collegamento di un plotter ad un microprocessore tramite un'interfaccia parallela. In molte esperienze, i fisici, i chimici, i medici o gli ingegneri etc. hanno bisogno della rappresentazione grafica di un segnale analogico. Ora, la ricezione di questo segnale e la sua visualizzazione su una periferica lenta non possono essere simultanee. Un dispositivo a microprocessore può risolvere questo problema nel modo seguente.

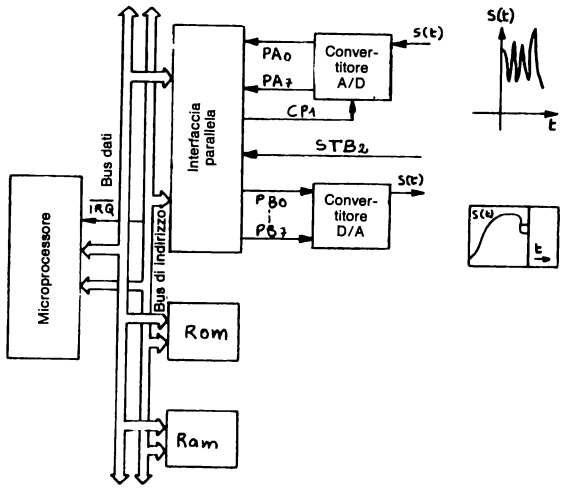


Figura 2 - *Collegamento di un microprocessore ad un plotter.*

Il segnale analogico S(t) è convertito in numero binario (8 bit) da un convertitore analogico/digitale (convertitore A/D). Questi numeri sono letti dal microprocessore tramite l'interfaccia, su 8 linee PA_0, PA_7 programmate come ingressi. Il segnale CP_1 assicura la sincronizzazione tra la lettura delle informazioni e la conversione di S(t).

Le informazioni sono provvisoriamente memorizzate nella RAM, via via che arrivano, e ne escono, su richiesta del microprocessore, tramite l'interfaccia parallela, sulle 8 linee PB_0, PB_7 programmate come uscite. Si dice che la RAM ha la funzione di "memoria tampone" (buffer in inglese), in quanto assorbe le discontinuità della trasmissione delle informazioni. Queste sono poi riconvertite in forma analogica, con un convertitore digitale/analogico (D/A), la cui uscita comanda il

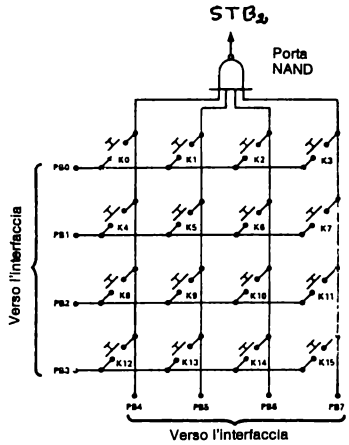


Figura 3a - Schema di una tastiera "non codificata".

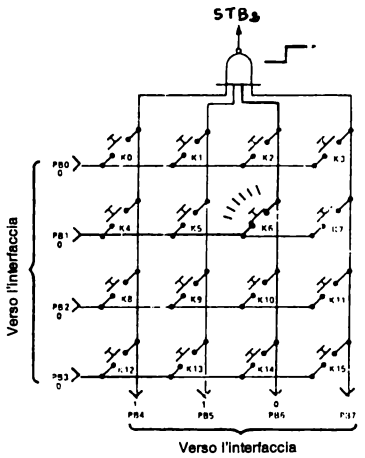


Figura 3c - Risultato della pressione su di un tasto.

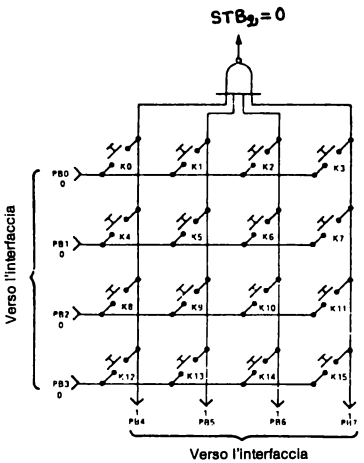


Figura 3b - Programmazione iniziale delle linee dei dati B dell'interfaccia.

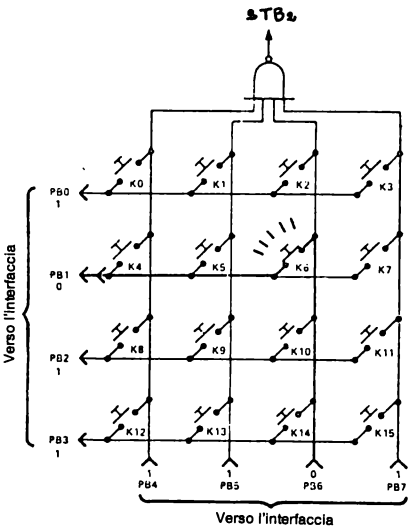
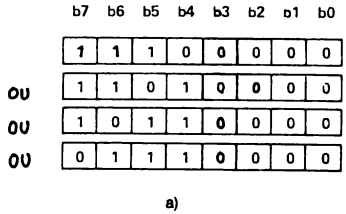


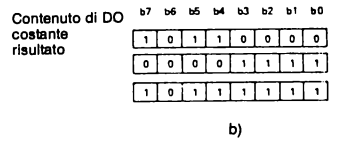
Figura 3d - Programmazione delle linee di dati B dell'interfaccia, nel senso inverso a quello della figura 3b.

plotter che disegna S(t). L'uscita di S(t) è comandata dal segnale STB_2, che causa un'interruzione della registrazione di S(t).

Collegamento di un microprocessore ad una tastiera

Un altro esempio di utilizzo di un'interfaccia parallela è dato nella figura 3. Si tratta di collegare una tastiera non codificata a 16 tasti ad un microprocessore. Lo schema della tastiera, paragonabile ad una matrice di quattro righe per quattro colonne, è disegnato nella figura 3a. Il collegamento al microprocessore è realizzato con metà di un'interfaccia parallela. Le righe della matrice sono collegate alle linee PB_0 - PB_3 e le colonne alle linee PB_4 - PB_7 (figura 3b). Inoltre le linee da PB_0 o PB_3 sono tenute da programma allo stato 0, e le linee da PB_4 a PB_7 allo stato 1.





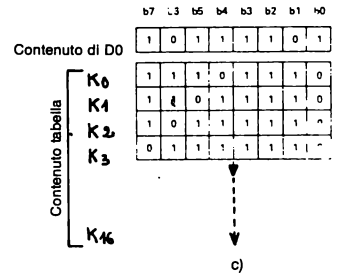


Figura 4a - I diversi possibili contenuti di D_0, dopo la lettura da parte del microprocessore, del registro dei dati B dell'interfaccia.

Figura 4b - OR logico tra il contenuto di D_0 e la costante 0000 1111.

Figura 4c - Tabella che consente, a partire dal contenuto di D_0, di determi-

Fino a quando non viene premuto un tasto, tutti gli ingressi della porta NAND sono allo stato 1, e l'uscita STB è a 1. Quando si preme un tasto (figura 3c) si mettono a contatto una riga e una colonna della matrice, facendo così passare a zero una linea da PB_4 a PB_7, e a 1 l'uscita della porta NAND. Questa transizione positiva, applicata all'interfaccia parallela sulla linea STB_2, richiede l'interruzione del microprocessore e il salto ad un sottoprogramma di identificazione dei tasti.

Il sottoprogramma di identificazione legge il contenuto del registro dei dati dell'interfaccia, associando alle linee PB_0 — PB_7 (figura 3c), e lo trasferisce in uno dei suoi registri dei dati, ad esempio D_0. Questo dato può assumere quattro valori, indicati nella figura 4a (le linee da PB_0 a PB_3, corrispondenti ai bit da 0 a 3, sono uscite allo stato 0). Nel nostro caso (figura 3c), il dato, trasferito in D_0, è 1011 0000. Il microprocessore esegue un OR logico tra il contenuto di D_0, e la costante 0000 1111 (figura 4b), e registra il risultato in D_0 quindi cambia la configurazione dell'interfaccia parallela, come indicato nella figura 3d, e invia il contenuto di D_0 sulle linee PB_0 - PB_7. Solo le linee PB_4 - PB_7, programmate come uscite, ne sono influenzate: legge il contenuto del registro dell'interfaccia, associato alle linee da PB_0 a PB_7, e lo trasferisce nel suo registro di dati D_0. Solo le linee PB_0 - PB_3,

programmate come ingressi, ne sono influenzate. Il contenuto di D_0 diventa allora: 1011 1101. Confrontandolo con i contenuti di una tabella, il microprocessore identifica il tasto premuto (figura 4c).

La chiusura di un contatto, dovuta alla pressione su di un tasto, avviene con dei rimbalzi che possono disturbare la sequenza di identificazione del tasto premuto. Questi rimbalzi durano in genere qualche millisecondo; basta quindi prevedere un ritardo, generato da software o da hardware, di circa un millisecondo, da quando viene ricevuta l'interruzione dalla tastiera, prima di lanciare la sequenza di identificazione del tasto.

L'identificazione di un contatto chiuso, tra molti, è un problema molto generale. La tastiera ne è solo un esempio. In una centrale di allarme la rivelazione degli incidenti avviene spesso con la chiusura di un contatto. In questo caso, il microprocessore identifica il ricevitore che ha generato l'allarme, e trasmette verso un posto di sorveglianza un codice che permette di riconoscere facilmente il ricevitore in questione.

Collegamento di un microprocessore ad un display a 7 segmenti

In molti dispositivi, come terminali, registratori, bilance elettroniche, strumenti di misura, etc. è necessario visualizzare un prezzo, un peso, una misura o il risultato di un calcolo. Si utilizzano allora dei display numerici a 7 segmenti, il cui funzionamento è spiegato chiaramente dalla figura 5.

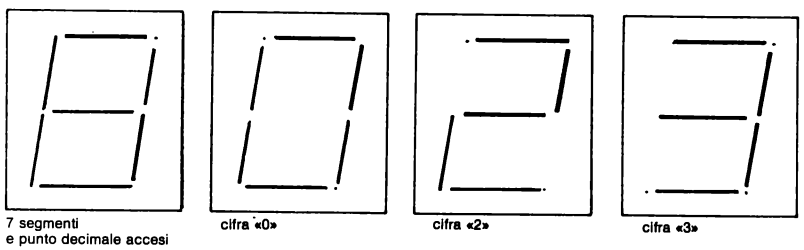


Figura 5 - *Esempi di rappresentazione di cifre su display a 7 segmenti.*

Un sistema di visualizzazione è comandato da un circuito di selezione dei display, e da un circuito che genera l'informazione da visualizzare. La figura 6 mostra un esempio di sistema a 16 cifre.

Il microprocessore invia, sulle linee PB_0 - PB_3 dell'interfaccia, le cifre da visualizzare codificate in BCD (Bynary Coded Decimal) (vedi figura 7), e, sulle linee PB_4 - PB_7, la posizione della cifra all'interno del sistema di visualizzazione. Poichè questa posizione è espressa in binario, ci vuole un decodificatore da 1 a 16 per selezionarla. Il valore della cifra, espresso in BCD, viene trasformato nella rappresentazione a 7 segmenti con un decodificatore BCD-7 segmenti.

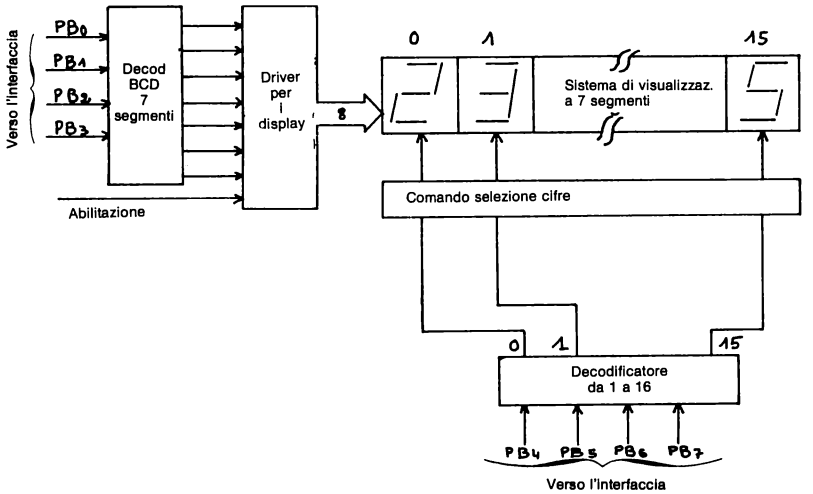


Figura 6 - *Sistema di visualizzazione a 16 cifre.*

0000	cifre	0
0001	cifre	1
0010	cifre	2
0011	cifre	3
0100	cifre	4
0101	cifre	5
0110	cifre	6
0111	cifre	7
1000	cifre	8
1001	cifre	9

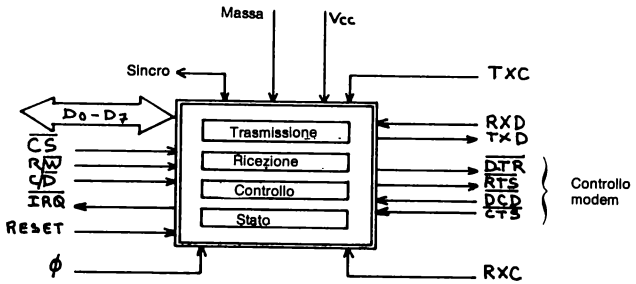
Per esprimere nove cifre in binario, ci vogliono 4 bit. Con 4 bit si possono però rappresentare 16 configurazioni diverse, di cui le prime dieci sono prese in considerazione e le altre sono proibite.

Figura 7 - *Le cifre decimali codificate in BCD.*

Per diverse ragioni di tipo tecnologico e per ridurre il consumo, il sistema di visualizzazione è "multiplexato", cioè le cifre vengono visualizzate una alla volta per 1/16 del tempo totale di visualizzazione (essendo 16 il numero dei display). Il microprocessore invia prima l'informazione al display 0, poi al display 1, e così di seguito fino al display 15. Dopo aver passato tutti i display, il microprocessore ricomincia la stessa operazione. Per produrre una immagine che si presenti stabile alla vista dell'uomo, la frequenza di questo ciclo non può essere inferiore a 50 Hz., e la frequenza del "multiplexaggio" deve raggiungere circa 1 kHz.

INTERFACCIA SERIE

In molte applicazioni, le periferiche trasmettono i dati su una linea e li ricevono da un'altra. Ogni carattere trasmesso o ricevuto da questo tipo di periferica è costituito da 5, 6, 7 o 8 bit. Può esserci un bit supplementare per consentire la rivelazione di un carattere errato (bit di parità pari o dispari). Si dice che un carattere è trasmesso in serie quando i bit del carattere non sono emessi simultaneamente, ma uno dopo l'altro. A seconda delle prestazioni richieste dalla trasmissione e dalla ricezione, la comunicazione può avvenire in modo asincrono o sincrono.



Nome dei piedini	Significato
D_0 — D_7	Bus dei dati
$\overline{CS}$	Linea di selezione dispositivo
R/$\overline{W}$	Linea di selezione lettura/scrittura
C/D	Linea di selezione del registro di controllo e di stato o del registro dei dati
IRQ	Richiesta di interruzione
RESET	Ritorno allo stato iniziale
0	Segnale di clock proveniente dal microprocessore
$\overline{RXC}$	Segnale di clock della ricezione
$\overline{CTS}$	Controllo della trasmissione tramite modem
$\overline{DCD}$	Controllo della ricezione tramite modem
$\overline{RTS}$	Richiesta di trasmissione da parte del trasmettitore
$\overline{DTR}$	Richiesta al modem di collegamento alla linea
TXD	Piedino di trasmissione
RXD	Piedino di ricezione
TXC	Clock di trasmissione
Vcc, massa	Alimentazione e massa
SYNCHRO	Rivelazione dei segnali di sincronizzazione, nella sincronizzazione interna, o dei comandi di sincronizzazione in quella esterna.

Figura 8 - *Organizzazione esterna e interna di un'interfaccia serie.*

I microprocessori operano su parole di 8, 12 o 16 bit in parallelo, e non possono utilizzare direttamente i caratteri trasmessi in serie dalle periferiche, nè inviarli a queste periferiche. Perciò è necessario utilizzare circuiti di interfaccia serie, che convertono una sequenza di carattere di 5, 6, 7 o 8 bit trasmessi in serie in una sequenza di caratteri di 8, 12 o 16 bit in parallelo, e viceversa. Alcuni circuiti di interfaccia possono funzionare in serie sia in modo sincrono che asincrono. Alcuni, più specializzati, funzionano o in modo sincrono o in modo asincrono. Torneremo su questi concetti più avanti, dopo aver studiato l'organizzazione esterna e interna dei circuiti di interfaccia serie.

Un circuito di interfaccia serie comprende linee di interfaccia con il microprocessore, linee di interfaccia con le periferiche e registri accessibili dal programmatore (figura 8).

Organizzazione esterna

Le linee di interfaccia con il microprocessore sono:
— la linea di selezione dispositivo: $\overline{CS}$;
— la linea di selezione di un gruppo di registri: C/$\overline{D}$ (C: gruppo di registri di controllo e di stato; $\overline{D}$: gruppo di registri di trasmissione e di ricevimento);
— la linea di selezione del modo lettura o scrittura: R/$\overline{W}$ permette di distinguere all'interno di un gruppo di registri quelli a sola lettura e quelli a sola scrittura);
— la linea di interruzione del microprocessore: $\overline{IRQ}$;
— la linea di ritorno allo stato iniziale: RESET.

Le linee di interfaccia con le periferiche sono:

— la linea di trasmissione: TXD;
— la linea del clock di trasmissione: TXC;
— la linea di ricezione: RXD;
— la linea del clock di ricezione: RXC;
— la linea SYNC che segnale, quando si opera in modo sincrono, che la sincronizzazione è stata effettuata.

Se l'interfaccia è destinata a comunicare tramite modem, ha, in più (figura 8):

— la linea di richiesta di collegamento del modem alla linea: $\overline{DTR}$;
— la linea di richiesta di trasmissione del tasmettitore: $\overline{RTS}$;
— la linea di controllo del trasmettitore da parte del modem: $\overline{CTS}$;
— la linea di controllo del ricevitore da parte del modem: $\overline{DCD}$.

Se l'interfaccia mette in comunicazione il microprocessore con l'unità a floppy disk (comunicazione sincrona), ha, inoltre, le linee di comando e di stato di questa unità.

Organizzazione interna

Il circuito di interfaccia comprende due gruppi di registri:
— registri dei dati (per la trasmissione e la ricezione);
— registri di controllo e di stato.

La selezione tra questi due gruppi di registri viene effettuata con la linea C/$\overline{D}$. La scelta tra i registri a sola lettura (registri di ricezione e di stato) e quelli a sola scrittura (registri di trasmissione e di controllo) si fa con la linea R/$\overline{W}$. Quando ci sono diversi registri di controllo, esiste in genere un registro di controllo primario e alcuni registri di controllo secondari. L'indirizzamento del registro di controllo primario è diretto. I registri di controllo secondari vengono indirizzati indirettamente tramite il registra primario.

I registri di controllo permettono di definire la configurazione funzionale dell'interfaccia, cioè il suo modo di funzionamento (asincrono, sincrono, SDLC), il formato del carattere (5, 6, 7 o 8 bit), la presenza o meno di bit di parità (pari o dispari), i caratteri di sincronizzazione ...

I registri di stato contengono informazioni sullo stato della trasmissione o della ricezione (trasmettitore vuoto, ricevitore pieno, errore di formato, errore di parità, sovraccarico).

Comunicazione serie asincrona

In questo modo di comunicazione, la trasmissione e la ricezione avvengono carattere per carattere. Un carattere comprende (figura 9) oltre ai bit del dato e al bit di parità, un bit di partenza (start bit) e uno o più bit di arresto (stop bit). Lo start bit permette al ricevitore di rivelare l'arrivo di un carattere, di sincronizzare il suo clock su questo bit e di assicurare una buona ricezione di ogni bit del carattere. Gli stop bit permettono di effettuare un controllo finale di buona ricezione. I bit di arresto possono essere 1,1 e 1/2, o 2. I bit di partenza e di arresto non contengono informazioni. Servono alla sincronizzazione, ma rallentano la comunicazione.

La *linea TXD* (figura 8) *del trasmettitore* (circuito di interfaccia) è a livello alto (marking line) quando non viene trasmesso nessun carattere. La trasmissione di un

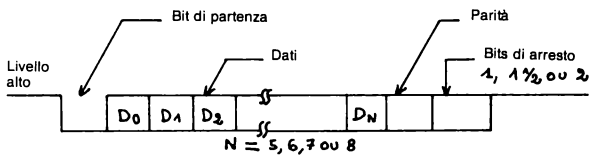


Figura 9 - *Una comunicazione serie asincrona avviene carattere per carattere. La rivelazione di un carattere e la sincronizzazione si effettuano sul bit di partenza. I bit di arresto permettono di effettuare un controllo finale di buona ricezione.*

carattere avviene, sotto il controllo del clock TXC del trasmettitore, inviando in successione su TXD un livello basso, corrispondente al bit di partenza, poi i bit del carattere propriamente detto, con o senza bit di parità, e infine il bit di stop. La linea TXD resta poi al livello alto (marketing line) fino a quando viene trasmesso un

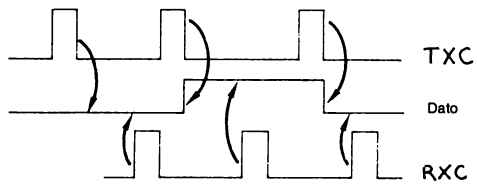


Figura 10 - *Principio della trasmissione e ricezione serie.*

nuovo carattere. I bit vengono emessi alla frequenza del clock, o a una frequenza 16 o 64 volte minore. La figura 10 descrive il principio di trasmissione alla frequenza del clock. L'intervallo di tempo tra due fronti di discesa del clock determina la durata di un bit.

La *linea RXD* (figura 8) *del ricevitore* (circuito di interfaccia) è allo stato alto quando non riceve nessun carattere. Il passaggio allo stato basso di questa linea per un intervallo di tempo pari almeno alla metà di un bit viene interpretato dal ricevitore come un bit di partenza, e provoca la ricezione di un carattere sotto il controllo dell'orologio di ricezione RXC. La ricezione avviene alla frequenza dell'orologio o a una frequenza 16 o 64 volte minore. Alla fine del carattere si trova ovviamente il bit di arresto. La figura 10 descrive il principio della ricezione alla frequenza dell'orologio. Alla fine di ogni carattere si trovano i bit di arresto allo stato 1. Dopo che il carattere è stato ricevuto, la linea di ricezione RXD resta allo stato alto fino a quando arriva un nuovo carattere.

Le comunicazioni serie asincrone sono utilizzate con le stampanti remote, le teletype e alcuni modem asincroni per le trasmissioni a lunga distanza. Il formato del carattere per una stampante remota è: 1 start bit, 5 bit di dato, 1 bit e 1/2 di stop (la durata di un bit di stop è una volta e mezzo la durata di un bit normale).

Il formato del carattere per teletype è di 11 bit: 1 bit di partenza, 7 bit di dato, 1 bit di parità e 2 bit di arresto.

Comunicazione serie sincrona

In questo modo di comunicazione (figura 11), i dati vengono trasmessi come un flusso continuo di bit, in cui non è possibile identificare l'inizio e la fine di un carattere.

È dunque necessario effettuare la sincronizzazione dei caratteri all'inizio del

blocco dei dati, al momento della loro ricezione. Questa sincronizzazione può essere esterna o interna.

La scelta tra questi due tipi di sincronizzazione deve essere fatta prima della ricezione, programmando opportunamente il circuito di interfaccia serie sincrono.

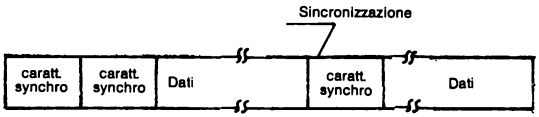
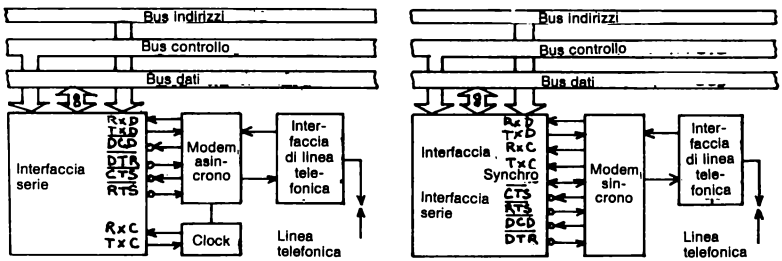


Figura 11 - *In una comunicazione serie sincrona, i dati vengono trasmessi con un flusso continuo di bit. La sincronizzazione viene fatta all'inizio del blocco dei dati, con uno o due caratteri.*

A) La sicnronizzazione interna viene ottenuta con il rilevamento, da parte del ricevitore (circuito di interfaccia) di uno o due caratteri di sincronizzazione all'inizio del messaggio. Il numero dei caratteri di sincronizzazione (1 o 2) è definito da programma al momento dell'inizializzazione dell'interfaccia.

Nella *sincronizzazione a un carattere*, il ricevitore cerca, bit per bit, una corrispondenza tra il dato ricevuto nel registro di shift e il contenuto del registro di sincronizzazione programmato al momento dell'inizializzazione. Se i due sono uguali significa che è stata stabilita la sincronizzazione, e che sarà mantenuta per tutto il blocco dei dati.

Nella *sincronizzazione a due caratteri*, il ricevitore cerca, bit a bit, il primo carattere di sincronizzazione, e poi il secondo. Se non riceve il secondo carattere, ricomincia la ricerca bit a bit del primo. I caratteri ricevuti prima che sia stabilita la sincronizzazione non vengono trasmessi al microprocessore.



Trasmissione asincrona su linea telefonica. *Trasmissione sincrona su linea telefonica.*

Nella trasmissione il modem converte i livelli 1 e 0, che provengono dall'interfaccia serie sulla linea TXD, in due frequenze per permettere le comunicazioni a grandi distanze. Al momento della ricezione viene fatta la conversione inversa.

Figura 12 - *Comunicazione tramite modem.*

B) Nel modo di sincronizzazione esterna, la ricezione comincia in genere sul fronte positivo di un impulso di clock RXC, dopo l'invio di un comando di sincronizzazione esterno su un piedino apposito dell'interfaccia (SYNCHRO).

Nella trasmissione, dopo l'eventuale invio dei caratteri di sincronizzazione, il dato parallelo, inviato dal microprocessore, passa in un registro di trasmissione (figura 8) del trasmettitore (circuito di interfaccia). Viene poi convertito da parallelo in serie e trasmesso sulla linea TXD sotto il controllo dell'orologio di trasmissione. I dati vengono emessi alla frequenza dell'orologio, secondo il principio della figura 10. Se il registro di trasmissione rimane vuoto, e non ci sono dati disponibili, il trasmettitore è "scarico", e viene inviato un carattere speciale sulla linea, in modo da mantenere la sincronizzazione. Questo carattere speciale, nel caso di sincronizzazione interna, può essere o un "Mark" (cioè un carattere con tutti i bit a 1), o il contenuto del registro di sincronizzazione.

Nella ricezione, dopo la sincronizzazione, i caratteri vengono ricevuti sul piedino di ricezione RXD, e trasmessi verso il microprocessore sotto il controllo del clock di ricezione RXC dell'interfaccia. Ogni carattere è convertito, nel circuito di interfaccia, da serie a parallelo, e, se non ci sono errori di trasmissione, viene letto dal microprocessore. I caratteri inseriti al momento della trasmissione, quando il trasmettitore era "scarico" (Mark o caratteri di sincronizzazione) possono essere eliminati da programma al momento della ricezione.

La figura 12 da due esempi di collegamento di interfaccie serie a dei modem, nel caso di trasmissione sincrona e asincrona.

Interfacce serie specializzate

In questo tipo di trasmissione i dati sono ricevuti e trasmessi nella forma serie asincrona. Il flusso dei dati contiene un carattere di sincronizzazione e di inizio del blocco dati, l'indirizzo del dispositivo a cui i dati sono destinati, informazioni di

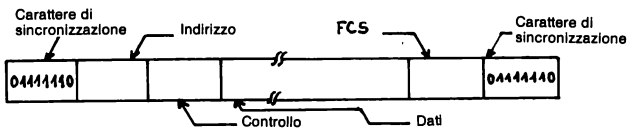


Figura 13 - *Secondo i protocolli SDLC/HDLC i dati sono ricevuti in forma seriale sincrona, la frequenza comprende un carattere di sincronizzazione, un indirizzo, alcune informazioni di controllo, i dati propriamente detti, una sequenza di controllo (FCS) e un ultimo carattere di sincronizzazione.*

controllo della sequenza, i dati propriamente detti, dei caratteri di controllo (FCS) e infine un carattere di sincronizzazione e di fine del blocco dati. I due caratteri di inizio e fine del blocco dati occupano un byte e contengono il valore 01111110. L'indirizzo è di 8 bit, e permette di indirizzare fino a 256 dispositivi.

Il campo dell'indirizzo può essere esteso da programma. La lunghezza della parola dei dati propriamente detti può essere definita da programma di 5, 6, 7 o 8

bit. Può essere introdotto, all'inizio del campo dei dati, un campo di controllo di lunghezza variabile. Il contenuto di questo campo è a disposizione dell'utente. Contiene i microcomandi che esso ha scelto. Il campo di controllo della sequenza contiene 16 bit generati dal trasmettitore. Al momento della trasmissione, il generatore di CRC (caratteri di controllo), contenuto nel trasmettitore, esegue al volo la divisione binaria della parte del flusso dei dati che comprende l'indirizzo, il campo di controllo e i dati, per un polinomio caratteristico: $X^{16} + X^{12} + X^5 + 1$. Il resto di questa divisione è scritto nei 16 bit che seguono la sequenza dei dati, nel campo FCS. Al momento della ricezione, il controllore di CRC esegue la stessa operazione sulla parte del flusso dei dati costituita da indirizzo, campo di controllo, dati e FCS. Se il risultato di questa divisione è uguale al valore fissato (FOB 8), il ricevitore ne deduce che non c'è errore di ricezione, altrimenti mette a 1 nel suo registro di stato un bit di errore, che segnala al microprocessore che la sequenza di dati ricevuta è errata.

Le comunicazioni che usano le procedure SDLC e HDLC collegano un sistema principale a dei terminali. Il dialogo avviene in modo diverso, a seconda che si abbia una configurazione isolata (stand-alone), a multi-terminali o ad anello chiuso.

— In una configurazione isolata, la comunicazione fra il sistema primario e quello secondario avviene direttamente.
— In una configurazione a multi-terminali, il dialogo è preceduto da una fase di indagine (polling), in cui il sistema principale cerca il terminale che ha chiesto la comunicazione.
— In una configurazione ad anello chiuso, il sistema primario (o principale) e i terminali (o sistemi secondari) sono su un anello sul quale circolano le informazioni. Il controllo dell'anello è affidato al sistema primario.

La figura 14 mostra un esempio di comunicazione su anello. Notiamo che questa configurazione è interessante per molte applicazioni: aereonautica, automobili, costruzioni... Questa tecnica permette di trasportare su un unico filo informazioni

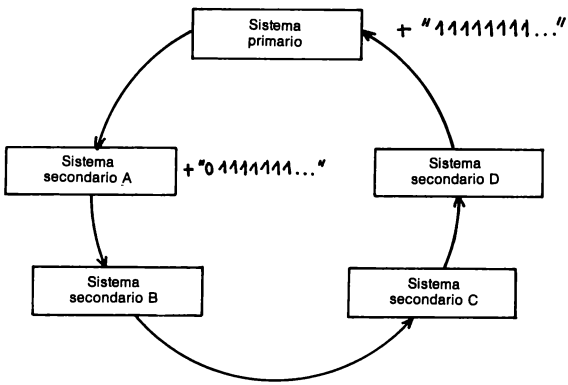


Figura 14 - *Esempio di comunicazione tramite anello.*

contenenti indirizzi e dati per diverse periferiche. In aereonautica questo significa una diminuzione del peso e un'economia del carburante. Nell'automobile una riduzione dei costi, in quanto diminuisce il numero dei collegamenti elettrici. Idem nelle costruzioni. In genere, per motivi di sicurezza, l'anello è duplicato, per evitare che il sistema si blocchi per eventuali guasti.

INTERFACCIA DI UNA UNITA' A FLOPPY DISK

L'utilizzo di un microcomputer richiede spesso la memorizzazione dei dati su di un supporto che conservi l'informazione dopo che si è tolta l'alimentazione. Esistono due supporti di questo genere: la cassetta megnetica e i floppy disk.

La registrazione sulla cassetta avviene sequenzialmente, per cui il tempo di accesso a un dato può risultare molto lungo, se si deve svolgere tutto il nastro. Su un floppy-disk, questo tempo è, in media, più breve, in quanto l'accesso è parzialmente diretto. Per questo, i floppy disk sono oggi preferiti alle cassette.

Come "formattare" il floppy-disk

Il floppy-disk, che ha le dimensioni di un disco audio a 45 giri, è ricoperto su entrambe le facce da una sostanza magnetica. È realizzato in materiale plastico flessibile, da cui il suo nome.

È prodotto da una busta (figura 15) anche quando è in funzione. In effetti, quando è introdotto nell'unità a lettura/scrittura, il disco è trascinato dall'esse del motore, e gira all'interno della sua busta. Può essere protetto dalla scrittura con un foro di protezione fatto sul disco dall'utente.

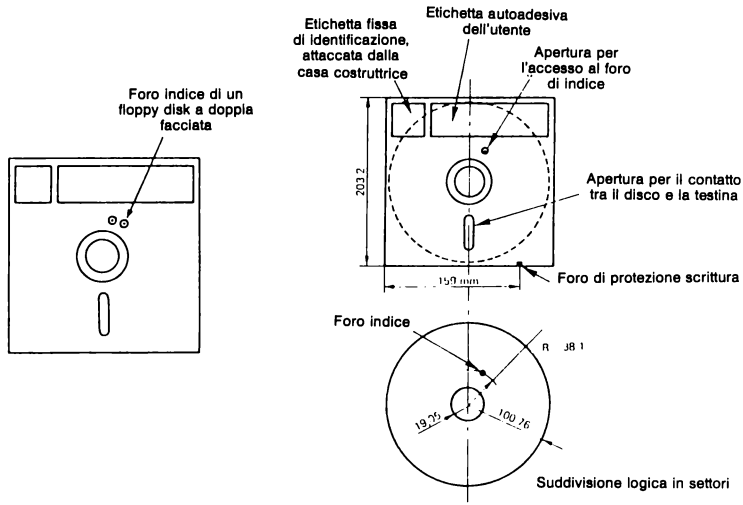


Figura 15 - *Presentazione di un floppy disk.*

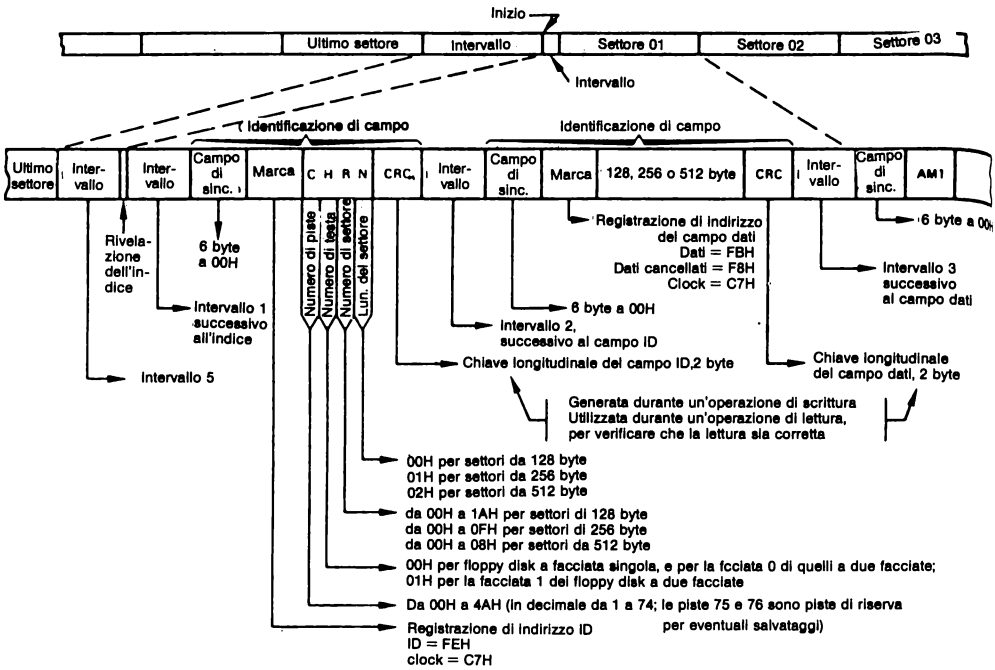


Figura 16 - *Formato di una pista.*

I dati sono registrati su cerchi concentrici che si chiamano piste. La testina di lettura/scrittura, che poggia direttamente sul disco, si sposta radialmente, all'interno della busta, da una pista all'altra, sotto l'azione di un meccanismo speciale pilotato da circuiti esterni, che si trovano nel controllore dell'unità. Le informazioni vengono registrate sequenzialmente, in modo sincrono, man mano che il disco gira all'interno della sua busta.

Un floppy-disk normale ha 77 piste su una sola faccia, se è a faccia singola, o su tutte e due, se è a due facciate. Le piste sono numerate dalla periferia verso il centro, da 00 a 76. Ogni pista forma un cerchio, in cui la fine si confonde con l'inizio. Ogni pista è suddivisa in un certo numero di settori di lunghezza fissa. A seconda di come viene "formattato" (operazione logica che consiste nel dividere una pista in settori), si possono avere 25, 15 o 8 settori per pista, rispettivamente di 128, 256 o 512 byte.

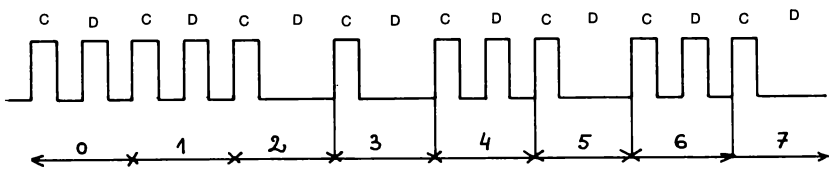
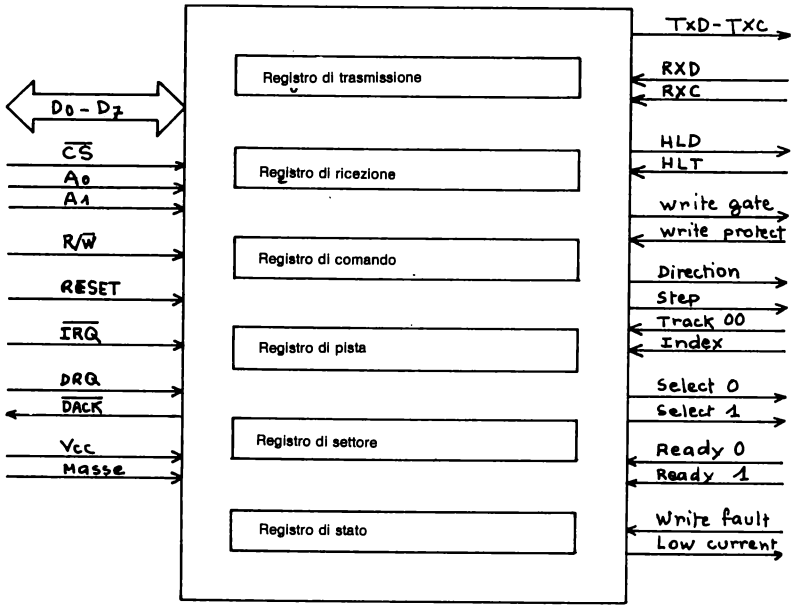


Figura 17 - *Alternanza dei bit di clock e dei bit di dato. Il byte di dato è 11001010.*



Nomi dei piedini	Significato
D_0 — D_7	Bus dei dati
$\overline{CS}$	Selezione dispositivo
A_0 — A_1	Selezione dei registri
R/$\overline{W}$	Selezione lettura/scrittura
RESET	Ritorno allo stato iniziale
$\overline{IRQ}$	Richiesta di interruzione
DRQ, $\overline{DACK}$	Richiesta di accesso diretto alla memoria e sua autorizzazione
V_{CC}, massa	Alimentazione e massa
TXD-TXC	Piedini di trasmissione
RDX-RXC	Piedino di ricezione e clock associato
HLD	Comando di posizionamento della testina sul disco
HLT	Segnalazione del posizionamento della testina sul disco
WRITE GATE	Autorizzazione per la scrittura
WRITE PROTECT	Protezione del disco dalla scrittura
DIRECTION, STEP	Senso di spostamento della testina, e passo dello spostamento
TRACK 00	Pista 0

Continuo tabella della pagina precedente	
INDEX	Inizio della pista
SELECT 0, SELECT 1	Selezione dell'unità 0 o dell'unità 1
READY 0, READY 1	Segnalazione che l'unità 0 o l'unità 1 è pronta
WRITE FAULT	Impossibilità di scrittura sul disco
LOW CURRENT	Diminuzione di corrente.

Figura 18 - *Organizzazione esterna e interna di un'interfaccia a floppy disk.*

L'inizio di una pista viene identificato con un foro che si trova sia sul disco che sulla busta. Ogni volta che il foro sul disco e quello sulla busta coincidono, una fotocellula riceve la luce emessa da un diodo, e genera un, impulso che indica l'inizio di una pista.

Ogni settore comprende (figura 16) un campo di identificazione ID (Identification Field) e un campo dati (Data Field), separati da intervalli (gap), che servono a compensare le variazioni di velocità del motore di trascinamento del disco. Questi intervalli vengono registrati sulle piste quando il disco è formattato. Il campo di identificazione definisce le caratteristiche del campo dati che lo segue, come: numero della testina (quando è a due facce), numero della pista, numero del settore, lunghezza in byte del campo dati. Ogni campo, sia di identificazione che di dati, comincia con una registrazione particolare (identificatore), che indica il tipo di campo (di indentificazione o di dati), e termina con due byte chiave (FCS). Questi due byte, risultato della divisione eseguita dal generatore di CRC al momento della registrazione, permettono, in fase di lettura di un settore, di controllare l'esattezza dei dati.

Un circuito di interfaccia per un'unità a floppy disk, pilotato da un microprocessore, può formattare i dischi. Durante le operazioni di scrittura, il generatore di CRC, che fa parte dell'interfaccia, calcola i due byte di chiave (FCS) e li scrive alla fine del campo. Durante le operazioni di lettura, il controllore di CRC esegue lo stesso calcolo, e confronta il risultato con i byte chiave. Se non c'è identità, l'interfaccia segnale al microprocessore la presenza di un errore.

La registrazione dei dati su una pista viene fatta in modulazione di frequenza (FM). Uno 0 è registrato con la frequenza del clock, un 1 con la frequenza doppia (figura 17). Ogni bit è preceduto da un impulso di clock; il bit propriamente detto viene poi registrato con un impulso per un 1, e nessun impulso per lo 0. Un byte comprende quindi 8 impulsi di clock, alternati con i bit del dato, che costituiscono il byte propriamente detto.

Gli "identificatori", cioè quelle registrazioni particolari che permettono di distinguere tra un campo di identificazione e un campo dati, devono essere diversi dalle informazioni che costituiscono i dati, memorizzate in ogni campo. Per far ciò, non sono registrati con gli impulsi di clock; nei loro byte non compaiono cioè impulsi di clock. D'altra parte, quando si formatta e si scrive il disco, l'interfaccia genera gli impulsi di clock necessari ad una corretta scrittura di questi identificato-

ri, e, al momento della lettura, provvede a ricostituire gli impulsi di clock necessari.

Gli impulsi di clock registrati sul disco servono solo a sincronizzare l'interfaccia all'unità a dischi. Perciò, in fase di lettura devono essere separati dai dati propriamente detti; d'altra parte, poichè i dati si presentano in serie, devono essere convertiti in parallelo dall'interfaccia, prima di essere inviati al microprocessore. Inversamente, al momento di una scrittura, dopo la conversione dei dati da parallelo a serie, l'interfaccia deve inserire, fra i bit dei dati, gli impulsi di clock.

L'organizzazione esterna e interna di un'interfaccia per una unità a dischi (figura 18), è quella di un'interfaccia sincrona, alla quale sono aggiunti piedini di comando e di stato dell'unità a dischi e dei registri specializzati: registro di comando dell'unità, registro del risultato del comando dopo la sua esecuzione, registro contenente il numero di pista su cui è posizionata la testina.

Organizzazione esterna dell'interfaccia

I piedini di collegamento tra il microprocessore e l'interfaccia dell'unità a floppy-disk sono identici a quelli di un'interfaccia serie classica:
D_0 — D_7: bus dei dati;
$\overline{CS}$: selezione dispositivo;
A_0 — A_1: selezione di uno di quattro registri;
R/$\overline{W}$: selezione lettura/scrittura;
$\overline{IRQ}$: richiesta di interruzione;
RESET: ritorno allo stato iniziale;
DRQ: richiesta di accesso diretto alla memoria;
$\overline{DACK}$: autorizzazione di accsso diretto alla memoria.

Ritorneremo ancora sulla funzione di questi ultimi due piedini, quando affronteremo la descrizione del circuito di accesso diretto alla memoria DMA.

Le linee di collegamento tra l'interfaccia e l'unità a dischi trasmettono o ricevono dei segnali che seguono le seguenti funzioni:

— *Trasmissione TXD-TXC*: invio degli impulsi di clock e dei dati dall'interfaccia verso il disco;

— *Ricezione RXD e impulsi del clock associato RXC*: l'unità a dischi fornisce un segnale composto che comprende gli impulsi di clock e i dati; un circuito disaccoppiatore inserito tra questa e l'interfaccia deve eliminare i segnali di clok dal dato;

— *Comando di posizionamento della testina sul disco HLD*: dà l'ordine di posizionare la testina sul disco e di preparare la trasmissione dei dati;

— *Posizionamento della testina HLT*: riceve un segnale, proveniente dall'unità a dischi, che indica all'interfaccia che il posizionamento della testina è stato

effettuato; l'intervallo di tempo che passa dal comando di posizionamento al posizionamento effettivo varia da un'unità a dischi a un'altra;

— *Autorizzazione alla scrittura WRITE GATE*: segnale di autorizzazione della scrittura quando la testina è posizionata;

— *Protezione dalla scrittura WRITE PROTECT*: quando questo ingresso è al livello alto, cioè quando l'utente ha protetto il disco dalla scrittura, l'interfaccia proibisce qualunque scrittura sul disco;

— *Direzione e passo*: i segnali forniti su questi due piedini indicano all'unità a dischi il senso di spostamento della testina (DIRECTION) e il passo di spostamento (STEP), pari allo spazio compreso tra due piste contigue; il numero di passi da effettuare è determinato per differenza tra il numero della pista su cui si trova la testina e quello della pista su cui deve andare;

— *Pista 0 (TRACK 00)*: indica all'interfaccia che la testina è posizionata sulla pista 0; questo segnale può essere utilizzato per la sincronizzazione, per l'inizializzazione e per la diagnostica;

— *INDEX*: indica l'inizio della pista;

— *Selezione SELECT 0 e SELECT 1*: il sistema di floppy disk collegato ad un microprocessore comprende due unità in un unico contenitore; la selezione di una delle due si fa con i piedini SELECT 0 e SELECT 1;

— *Indicazione di buon funzionamento READY 0 e READY 1*: è inviato all'interfaccia quando l'unità a dischi attiva è pronta a leggere o a scrivere; il segnale, fornito su uno di questi due piedini è il risultato di un certo numero di test, tra cui quello di chiusura della porta dell'unità;

— *Errore di scrittura WRITE FAULT*: segnala all'interfaccia che non si può scrivere sull'unità a dischi;

— *Diminuzione di corrente LOW CURRENT*: uscita attivata ogni volta che il numero della pista è maggiore di 43.

Organizzazione interna dell'interfaccia

Un'interfaccia per unità a dischi comprende in genere i seguenti registri:

— *Il registo di trasmissione* che assicura, tra l'altro, la conversione parallelo/serie dei dati che provengono dal microprocessore;
— *Il registro di ricezione* che effettua, tra l'altro, la conversione serie/parallelo dei dati provenienti dall'unità a dischi;

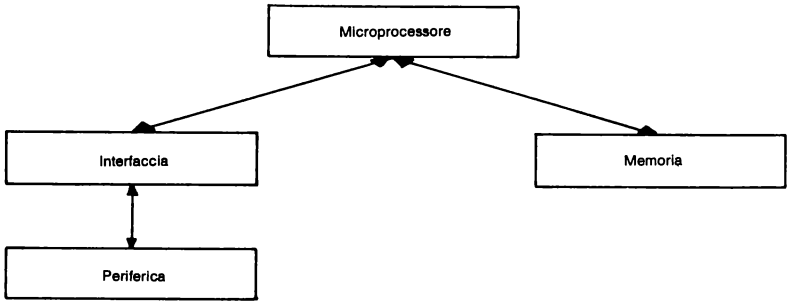


Figura 19 - *Collegamento di una periferica alla memoria tramite il microprocessore.*

— *Il registro di controllo o di comando*: contiene praticamente una macroistruzione, che viene decodificata dall'interfaccia e quindi convertita in microcomandi, eseguiti dall'unità a dischi;
— *Il registro di pista*: contiene il numero della pista su cui si trova la testina;
— *Il registro di settore*: contiene il numero del settore su cui è in corso un'operazione di lettura o scrittura;
— *Il registro di stato*: contiene il risultato dell'esecuzione di un comando (buono o cattivo, e, in quest'ultimo caso, gli errori) e lo stato dell'unità.

I comandi dell'unità a floppy-disk

Sono contenuti nella memoria di programma, per essere inviati, tramite il microprocessore, all'interfaccia. Quest'ultima decodifica ogni comando, e genera i segnali di controllo per l'unità a dischi.

I comandi permettono lo spostamento della testina verso una certa pista (SEEK), la lettura (READ) e la scrittura (WRITE) dei dati in uno o più settori, il formattamento del disco (FORMAT) ...

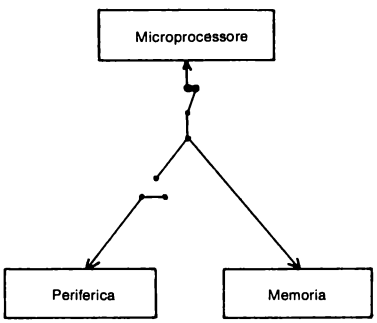


Figura 20a - Funzionamento normale (collegamento microprocessore-memoria).

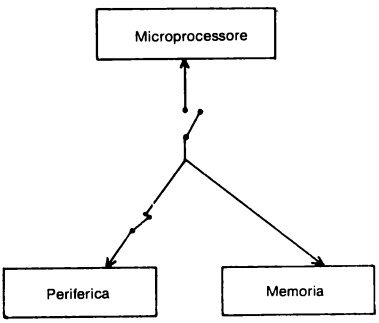
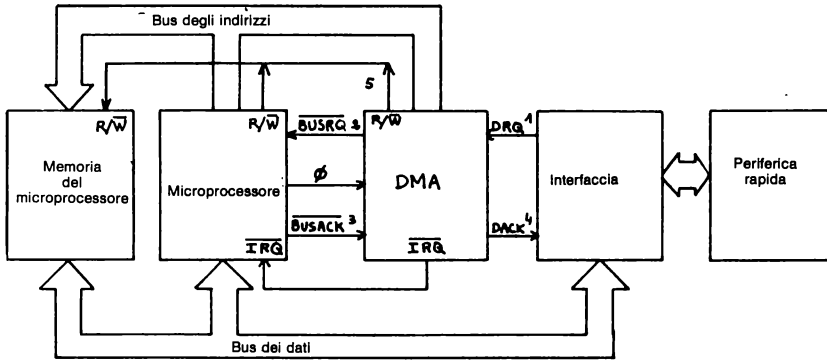


Figura 20b - Funzionamento in DMA (collegamento periferica-memoria).

Figura 20 - *Principio di funzionamento dell'accesso diretto alla memoria.*



1 richiesta di accesso diretto alla memoria da parte della periferica
2 trasmissione di questa richiesta al microprocessore
3 autorizzazione di accesso diretto alla memoria da parte del microprocessore
4 trasmissione dell'autorizzazione alla periferica

Figura 21 - *Collegamento di una periferica veloce a un microprocessore in DMA.*

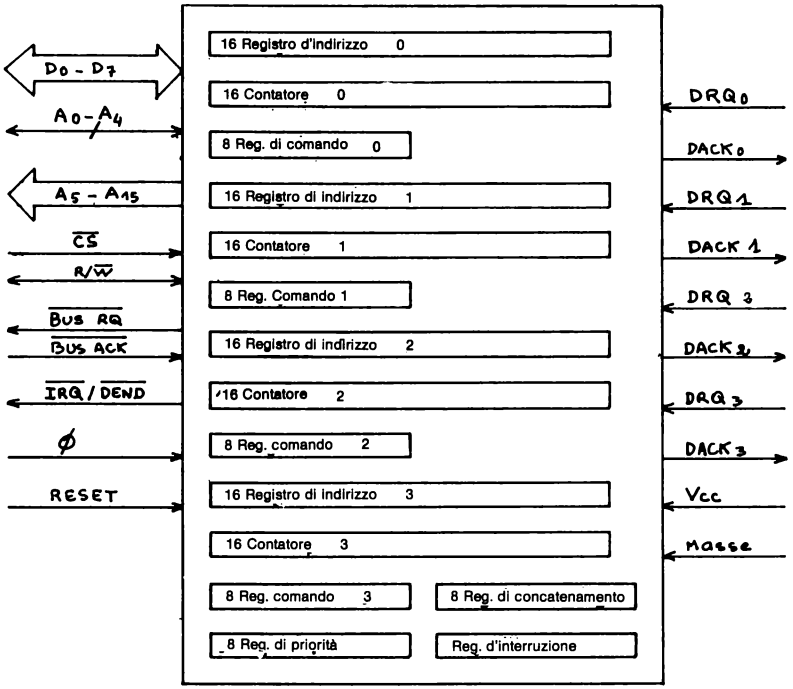
CIRCUITO DI ACCESSO DIRETTO ALLA MEMORIA (DMA)

I trasferimenti di dati tra una periferica e le memorie avvengono in genere, attraverso il microprocessore (figura 19). Quando le periferiche veloci, come un'unità a dischi o un elaboratore, possono però accedere alla memoria di un sistema direttamente, senza passare dal microprocessore, che è troppo lento. Questo è ciò che si chiama accesso diretto alla memoria.

Durante il normale funzionamento, le vie di comunicazione (bus dei dati, bus degli indirizzi, bus di controllo) tra il microprocessore e la memoria sono aperte, e quelle tra la memoria e la periferica veloce sono chiuse (figure 20a). Quando la periferica chiede un accesso diretto alla memoria, viene interrotto il collegamento tra il microprocessore e la memoria, e si realizza invece quello tra la periferica e la memoria (figura 20b). Il dispositivo che gestisce l'accesso diretto alla memoria si chiama circuito DMA.

Programmabile dal microprocessore, il DMA esegue, una dopo l'alta, le seguenti funzioni (figura 21):

— Ricezione, sul piedino DRQ, delle richieste di accesso diretto che sono poi trasmesse al microprocessore sul piedino $\overline{\text{BUSRQ}}$;
— Emissione verso la periferica, sul piedino DACK, del segnale di autorizzazione della richiesta di DMA, emesso dal microprocessore sul piedino $\overline{\text{BUSACK}}$;
— Gestione dell'indirizzamento della memoria per tutta la durata del trasferimento del blocco dei dati tra la periferica e la memoria. Il trasferimento avviene sotto il controllo del clock 0 del microprocessore. La fine del trasferimento viene segnalata al microprocessore sul piedino di interruzione $\overline{\text{IRQ}}$. Dal momento dell'autorizzazione del DMA, i bus dei dati, degli indirizzi e di controllo del



Nome dei piedini	Significato
D_0 — D_7	Bus dei dati
A_0 — A_3	Bus degli indirizzi della memoria (uscita) linee di selezione di un registro (ingresso)
A_4 — A_{15}	Bus degli indirizzi della memoria
$\overline{CS}$	Selezione dispositivo
$R/\overline{W}$	Selezione lettura/scrittura
$\overline{BUSRQ}$	Richiesta di accesso diretto alla memoria
$\overline{BUS\ ACK}$	Accettazione da parte del microprocessore della richiesta di accesso diretto
$\overline{IRQ}/\overline{DEND}$	Richiesta di interruzione/fine del trasferimento in DMA
0	Clock dei trasferimenti in DMA
RESET	Ritorno allo stato iniziale
DRQ_0, DRQ_1, DRQ_2, DRQ_3	Richiesta di accesso diretto alla memoria dalle periferiche 0, 1, 2 e 3

Continuo tabella della pagina precedente	**Significato**
$DACK_0$, $DACK_1$, $DACK_2$, $DACK_3$	Autorizzazione del DMA per le periferiche 0, 1, 2 o 3
Vcc, massa	Alimentazione e massa

Figura 22 - *Organizzazione, esterna e interna di un circuito DMA.*

microprocessore passano allo stato di "alta impedenza", scollegando di fatto il microprocessore, dai bus del sistema.

Il trasferimento tra la periferica veloce e la memoria può avvenire in due modi:

— *a raffica*, cioè per trasferimento di un blocco completo di dati;
— *in cycle stealing*, cioè "rubando" un ciclo di memoria, e trasferendo un byte ad ogni richiesta di accesso diretto. In questo caso, la periferica, quando riceve l'autorizzazione, rimanda la richiesta sul piedino DRQ.

Vediamo in dettaglio l'organizzazione esterna e interna di un circuito di DMA (figura 22). Esso contiene, in genere, quattro canali DMA. Ad ogni canale sono associati tre registri specializzati (figura 22):
— *un puntatore o registro di indirizzo* che punta la zona di memoria in cui i dati saranno scritti o letti;
— *un contatore* che contiene il numero di byte da trasmettere;
— *un registro di comando* che definisce il tipo di operazione DMA da eseguire: trasferimento a raffica o in cycle stealing, senso del trasferimento,...

Inoltre esistono due registri coinvolti in tutti i modi di funzionamento del DMA:
— *un registro di comando di priorità*, che può abilitare o disabilitare ogni funzione del DMA, e che seleziona il funzionamento a priorità fissa o a priorità circolare. Nel primo caso la priorità va da 0 a 3, e se ci sono più richieste di DMA contemporaneamente, viene presa in considerazione quella con il livello più alto. Nel funzionamento a priorità circolare, l'ultima periferica che ha avuto un accesso alla memoria in DMA passa al livello di priorità più basso.

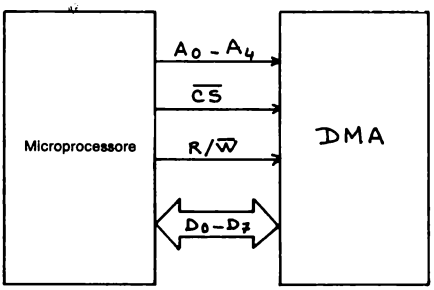


Figura 23 - *Piedini coinvolti nell'inizializzazione di un circuito DMA.*

— *un registro di stato o di interruzione*, che permette al microprocssore, al momento di un'interruzione, di riconoscere se è stato il circuito DMA a generare l'interruzione, e, in questo caso, di conoscere la funzione DMA che l'ha generata.

Un registro di concatenamento permette di realizzare più trasferimenti in DMA su un unico canale, conservando così la lettura o la scrittura ripetuta di una certa parte della memoria. Il concatenamento si può effettuare in genere solo su tre canali (per esempio i canali 0,1 e 2) mentre i registri del canale 3 servono da memorie tampone. La correttezza del concatenamento dei dati e la scelta del canale su cui si opera il concatenamento sono indicati dal registro di concatenamento.

Inizializzazione del circuito DMA

La figura 23 mostra i piedini che servono per l'inizializzazione del circuito DMA sotto il controllo del microprocessore.

La preparazione di un canale per un trasferimento in DMA consiste nelle seguenti tre operazioni:

— caricamento dell'indirizzo di inizio nel registro di indirizzo;
— caricamento del numero di byte da trasferire nel contatore;
— programmazione del registro di comando in base alle caratteristiche del trasferimento.

Stabilita la configurazione del canale, la domanda di trasferimento dei dati viene autorizzata posizionando opportunamente un bit del registro di comando della priorità, e l'eventuale bit di convalida dell'interruzione. Se è necessario un concatenamento dei dati per quel canale, si programmerà opportunamente il registro di concatenamento.

TEMPORIZZATORE/CONTATORE DI EVENTI

In numerose applicazioni industriali, è necessario generare delle temporizzazioni o contare degli oggetti. Nell'industria chimica, per esempio, la miscela di diversi liquidi richiede versamenti successivi a intervalli di tempo ben determinati; la regolamentazione del traffico ad un incrocio richiede il conteggio delle vetture sulle diverse strade che vi confluiscono, e il mantenimento del verde o del rosso su ognuna di esse per un tempo che dipende dal traffico.

Il microprocessore può gestire tutte queste funzioni, ma mentre ne esegue una non fa niente altro. Da questo nasce l'idea di sottrargli compiti "subalterni", e affidarli a circuiti specializzati.

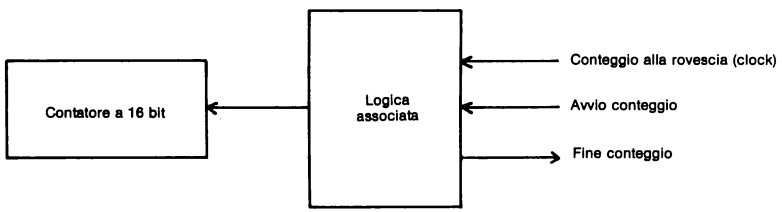
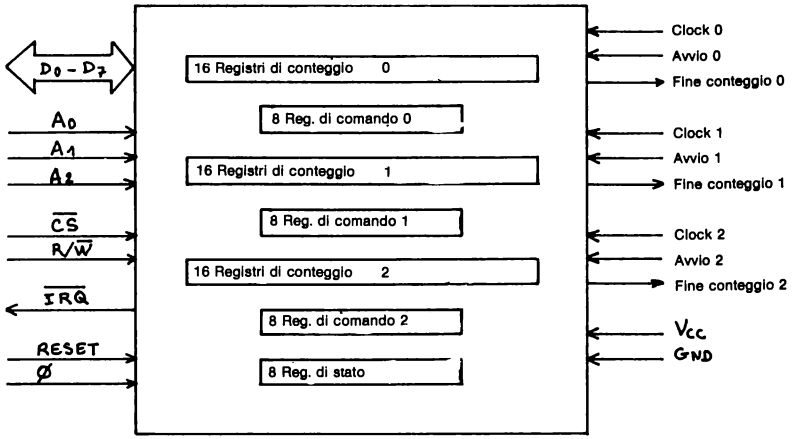


Figura 24 - *Funzione di conteggio alla rovescia.*



Nome dei piedini	Significato
D_0 — D_7	Bus dei dati
A_0, A_1, A_2	Selezione di un registro
$\overline{CS}$	Selezione del circuito
R/W	Selezione lettura/scrittura
$\overline{IRQ}$	Richiesta di interruzione
RESET	Ritorno allo stato iniziale
0	Clock
V_{CC}, GND	Alimentazione e massa
Clock 0, 1, 2	Clock esterni
Avvio 0, 1, 2	Piedini di avvio conteggio
Fine conteggio 0, 1, 2	Piedini di fine conteggio

Figura 25 - *Organizzazione esterna e interna di un temporizzatore.*

Un circuito di questo genere comprende in genere tre canali di temporizzazione-/conteggio degli eventi. Ogni canale comprende un contatore a 16 bit, una logica associata a tre piedini: un piedino di conteggio alla rovescia, un piedino di avvio del conteggio, e un piedino di fine conteggio (figura 24). Il clock del conteggio è interno (clock del microprocessore) o esterno; può funzionare in modo asincrono (ogni volta che avviene un evento esterno) o sincrono (impulsi di clock generati a una frequenza determinata). Il conteggio può cominciare o da quando il contatore è caricato con la quantità opportuna, o da quando un impulso esterno è applicato al suo piedino di avvio conteggio. Quando il contatore arriva a zero, cioè quando la temporizzazione richiesta è terminata, o quando il numero di eventi da contare è stato raggiunto, il canale lo segnala al microprocessore sul piedino $\overline{IRQ}$, e alla periferica sul piedino "fine conteggio", associato a ciascun canale.

L'organizzazione esterna e interna di un temporizzatore/contatore di eventi è

descritta nella figura 25. Ogni canale ha a disposizione i seguenti registri programmabili:

— *registro di conteggio a 16 bit*, in cui il microprocessore carica la quantità da decrementare;
— *registro di comando di ogni funzione*, nel quale il microprocessore definisce la funzione del circuito. A seconda del contenuto di questo registro, il circuito può eseguire una delle seguenti funzioni: temporizzazione, conteggio di eventi, generazioni di impulsi (ad esempio, segnali di clock con una certa frequenza); può funzionare in risposta ad un segnale sul piedino di avvio conteggio, o no.

Inoltre esiste, in genere, un registro di stato, comune a tutti i canali. L'analisi del contenuto di questo registro permette al microprocessore, quando arriva una richiesta di interruzione, di determinare se questa domanda viene dal temporizzatore/contatore di eventi, e, in questo caso, il canale responsabile.

INTERFACCIA DI UN VIDEO

L'utilizzo di un microprocessore richiede la visualizzazione dei dati, che si può ottenere o su display alfanumerici o su un tubo catodico.

La visualizzazione su display alfanumerici si effettua, in genere, tramite un'interfaccia parallela (vedi pag. 79), e occupa dal 10% al 20% del tempo di elaborazione del microprocessore, per visualizzare una linea di 16 caratteri. Presenta d'altra parte il vantaggio di essere relativamente economica, in quanto è realizzata via software, e richiede pochissimo hardware. Se si vogliono, però, visualizzare 16 linee di 64 caratteri, o 24 linee di 80 caratteri, non è più possibile utilizzare i display, a causa del loro costo e della eccessiva complessità dei circuiti necessari.

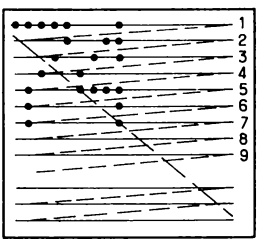


Figura 26 - *Scansione dello schermo.*

In questo caso, la soluzione migliore è un tubo catodico, simile a un tubo TV, sul quale un raggio luminoso traccia una serie di linee (figura 26).

Applicando impulsi luminosi alla griglia di comando del tubo catodico, in determinati istanti durante la scansione dello schermo, possono essere generati dei caratteri. La figura 27 mostra lo schema di principio di un sistema di visualizzazione in cui l'interfaccia fornisce tutta la logica di comando. Questa è, in genere, programmabile: si possono scegliere il numero di righe per una fila di caratteri, e il numero di file.

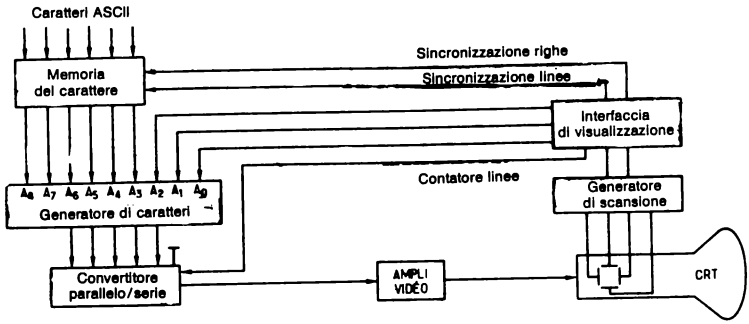


Figura 27 - *Schema semplificato di un sistema di visualizzazione.*

Principio di funzionamento

Il procedimento di scrittura di un carattere formato da una matrice di 35 punti (7x5) è mostrato nella figura 28. Il microprocessore trasmette alla memoria dei caratteri, associata all'interfaccia, un carattere ASCII (1), sotto forma di 7 bit di indirizzo da A_3 a A_9. Il generatore di caratteri li riceve e, a seconda del valore di A_0, A_1, A_2, invia, dopo una conversione parallelo/serie, alla griglia del tubo catodico, una parola di 5 bit, corrispondente ad una riga della matrice del carattere da visualizzare. Poichè nel nostro caso, un carattere è formato da sette righe di cinque punti ognuna (figura 28), l'indirizzo da A_3 a A_9 deve essere applicato sequenzialmente sette volte al generatore di caratteri in sincronismo con la scansione delle righe mentre l'indirizzo da A_0 a A_2 passa da zero a sei.

Nella figura 29, che mostra uno schermo di 16 file di 64 caratteri di 35 punti ognuno, compare un cursore che determina in ogni istante la posizione del nuovo carattere. Una fila di caratteri si ottiene nel modo seguente (figura 29). Il primo carattere ASCII, corrispondente ad "A" viene inviato al generatore di caratteri.

La parola di 5 bit 00100, corrispondente a $A_0 = A_1 = A_2 = 0$, viene trasmessa dal generatore di caratteri al convertitore parallelo/serie. Tutti i bit della parola sono poi applicati uno dopo l'altro alla griglia del tubo catodico sotto il controllo dell'interfaccia, in modo che i cinque bit della prima riga del carattere sono visualizzati nella scansione della prima linea. Nella "A" della figura 29 solo il bit centrale è a 1, per cui appare un solo punto centrale.

Il secondo carattere ASCII, corrispondente a "B" fornito dalla RAM dei dati sotto il controllo dell'interfaccia di visualizzazione, serve come indirizzo per il generatore di caratteri, che trasferisce verso lo schermo i 5 bit 11110 della prima riga di "B". Questi bit sono visualizzati nella scansione della prima linea, spostati, nel nostro caso, di tre bit rispetto al primo carattere. Il processo continua finchè la scansione della prima riga è completata.

Il generatore di scansione orizzontale, sotto il controllo dell'interfaccia di visualizzazione, produce un segnale di ritorno, e poi inizia la scansione della linea 2($A_0 = 1$, $A_1 = 0$, $A_2 = 0$), cioè la seconda riga di A, B, C, D, ... W, X, Z.

La stessa sequenza viene ripetuta finchè la prima riga di caratteri (7 linee di scansione) è completamente visualizzata. Dopodichè il punto luminoso percorre, sotto il controllo dell'interfaccia di visualizzazione, 5 linee di scansione, su cui non è effettuata alcuna operazione di visualizzazione. Questo serve a generare una interlinea tra due righe di caratteri, e anche a visualizzare un cursore.

Dopo l'interlinea, la sequenza di visualizzazione della seconda riga di caratteri si svolge in maniera analoga alla prima, e così di seguito per 16 righe di 64 caratteri. Questa operazione è ripetuta ogni 20 millisecondi, per garantire la permanenza del carattere sullo schermo.

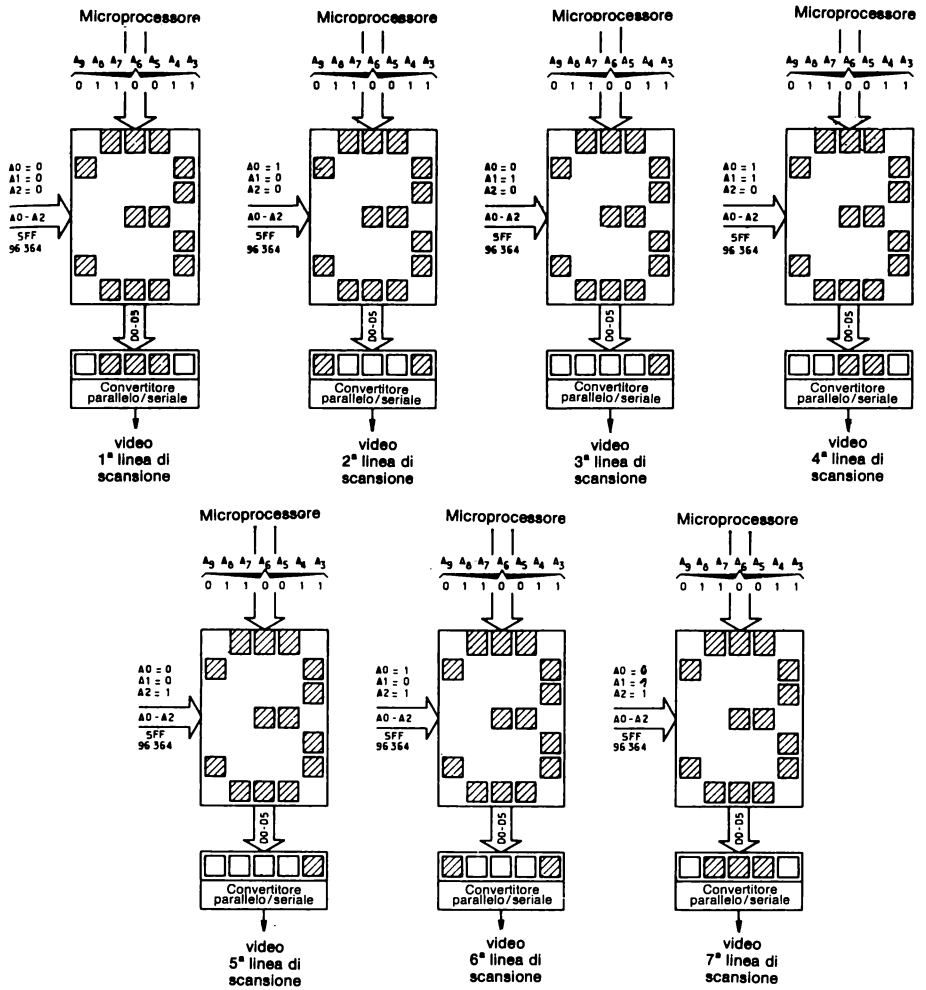


Figura 28 - *Generazione di un carattere da visualizzare.*

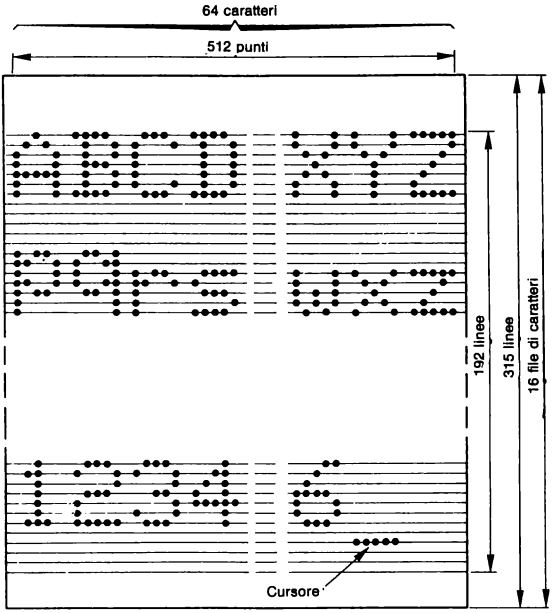


Figura 29 - *Metodo di visualizzazione su schermo.*

Sullo schermo invece, oltre ai caratteri, un cursore che segnala la posizione del nuovo carattere da visualizzare. la gestione di questo cursore viene fatta tramite l'interfaccia di visualizzazione, che riceve tre tipi di istruzioni da parte dell'operatore:

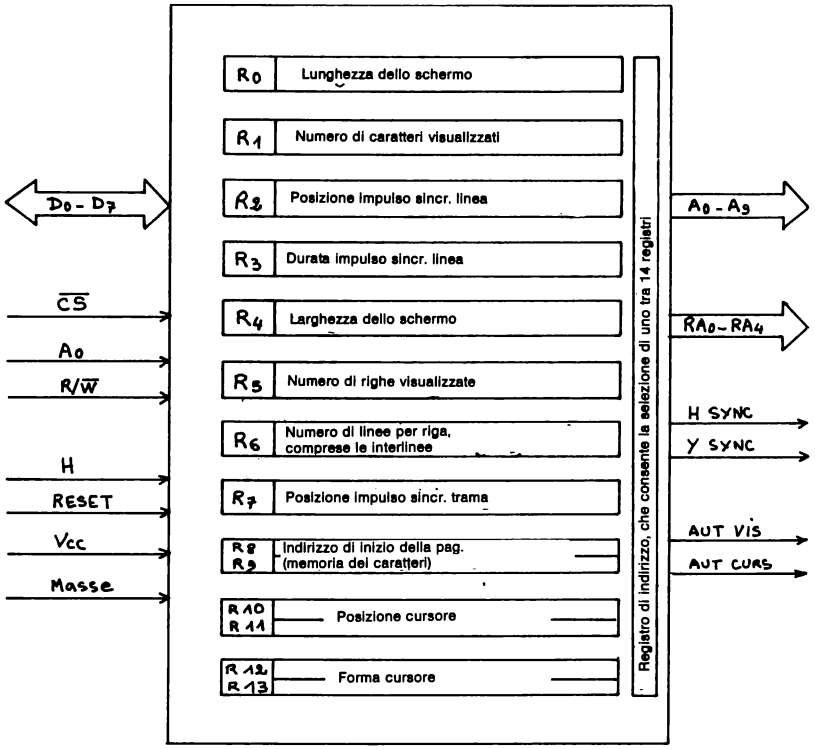
— *cambiamento della posizione del cursore sullo schermo, senza modifiche del testo*;
— *cancellazione di una parte o di tutto lo schermo*;
— *avanzamento di un posto a destra del cursore*; operazione effettuata automaticamente ogni volta che si visualizza un nuovo carattere.

Alla fine della pagina, cioè quando lo schermo contiene 16 righe di 64 caratteri, l'interfaccia di visualizzazione evita il ritorno del cursore in alto, e impedisce la scrittura di un nuovo testo sopra quello vecchio. Blocca il percorso del cursore sull'ultima riga e sposta in alto progressivamente il testo. Le nuove righe si inseriscono dal basso, le righe superiori sono perse.

Organizzazione esterna e interna

L'interfaccia di visualizzazione è, in genere, programmabile. Dispone di diversi registri che permettono di definire i diversi parametri della visualizzazione (figura 30):

— lunghezza in caratteri dello schermo (Registro R0);



Nome dei piedini	Significato
D_0 — D_7	Bus dei dati
$\overline{CS}$	Selezione dispositivo
A_0	Selezione registro di indirizzo/registro di comando
$R/\overline{W}$	Selezione del modo lettura/scrittura
H	Clock dei caratteri
RESET	Ritorno allo stato iniziale
V_{CC}, massa	Alimentazione e massa
A_0 — A_9	Indirizzo della memoria di visualizzazione (pagina)
RA_0 — RA_4	Numero di linea sotto scansione, all'interno di una fila, comprese le interlinee
H SYNC	Impulso di sincronizzazione linea
V SYNC	Impulso di sincronizzazione trama
AUT VIS	Autorizzazione di visualizzazione
AUT CURS	Autorizzazione di visualizzazione del cursore

Figura 30 - *Organizzazione esterna e interna di una interfaccia di visualizzazione.*

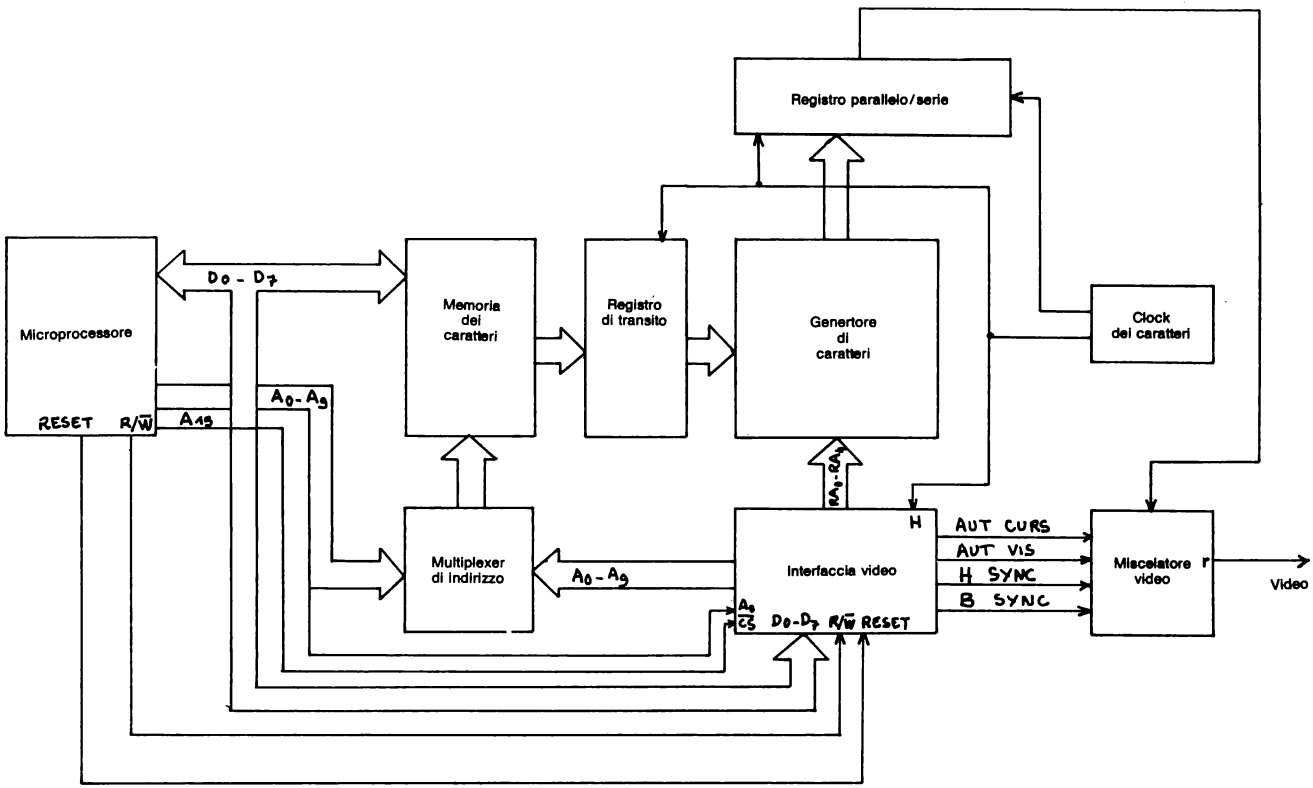


Figura 31 - *Schema logico di un sistema di visualizzazione.*

— numero di caratteri per fila (Registro R1);
— posizione dell'impulso di sincronizzazione di linea in numero di caratteri;
— durata dell'impulso di sincronizzazione di linea (Registro R3);
— larghezza dello schermo in numero di file di caratteri (Registro R4);
— numero di file visualizzate (Registro R5);
— numero di linee per fila, comprese le interlinee (Registro R6);
— posizione dell'impulso di sincronizzazione della trama (Registro R7);
— inizio della memoria di visualizzazione (Registro R8 e R9);
— posizione del cursore (Registro R10 e R11);
— forma del cursore (Registro R12 e R13).

Il posizionamento di un testo su di uno schermo richiede, generalmente, di lasciare dei margini a sinistra, a destra, in alto e in basso. Il margine a sinistra, il numero di caratteri visualizzati e il margine a destra, espressi in numero di caratteri, determinano la capacità massima di caratteri visualizzabili su una fila. Per motivi di estetica, il margine di sinistra è uguale al margine di destra. Quando si ottiene programmando in modo opportuno l'istante del "ritorno-linea". Analogamente, il margine in alto, il numero di file di caratteri visualizzate e il margine in basso danno il numero massimo di file visualizzabili e l'altezza dello schermo. Per avere margini uguali, in alto e in basso, si programma la posizione d'inizio dell'impulso di ritorno-trama.

La pagina di caratteri da visualizzare sullo schermo è contenuta nella memoria di visualizzazione, il cui inizio è indicato nella interfaccia di visualizzazione per mezzo di un registro speciale (R8 - R9).

Infine, la definizione della forma del cursore si fa programmando in due registri speciali, la linea di inizio e la linea di fine del cursore. La posizione del cursore sullo schermo è indicata da altri due registri.

La programmazione di un registro di comando si effettua in due tempi:
— scrivendo nel registro di indirizzo il numero del registro di comando in questione ($A_0 = 0$, $R/\overline{W} = 0$, $\overline{CS} = 0$);
— scrivendo nel registro di comando in questione il comando ($A_0 = 1$, $R/\overline{W} = 0$, $\overline{CS} = 0$).

La selezione del registro di comando e il comando propriamente detti sono trasferiti, sul bus dei dati D_0 — D_7, dalla memoria di programma tramite il microprocessore.

La sincronizzazione tra la console di visualizzazione e l'interfaccia di visualizzazione si realizza tramite il clock di carattere H.

La visualizzazione di una pagina richiede (figura 31):

— la scansione della memoria dei caratteri mediante A_0 — A_9;
— la scansione del generatore di caratteri tramite i dati inviati dalla memoria dei caratteri e la selezione delle righe di una fila RA_0 — RA_4;
— la scansione dello schermo sotto il controllo dei segnali H Sync, V Sync, Aut Vis e Aut Curs.

CAPITOLO 4

I MODI DI INDIRIZZAMENTO

I modi di indirizzamento caratterizzano le possibilità di accesso del microprocessore alle informazioni che si trovano in memoria. Per capirli meglio ci serviremo di alcune analogie.

INDIRIZZAMENTO ESTESO

Prendiamo l'esempio di un libro. Il metodo più semplice per accedere al contenuto di una pagina è di cercare il numero di quella pagina. Il microprocessore opera in questo stesso modo, quando legge o scrive in una locazione di memoria con il modo di indirizzamento esteso.

La figura 1 mostra un esempio di indirizzamento in modo esteso.

Nel corso di un programma, il microprocessore legge, per esempio, nella locazione 5000 della memoria di programma, l'istruzione ST D0, che significa: memorizza il contenuto del registro D0 all'indirizzo che segue. Nelle locazioni 5001 e 5002, trova l'indirizzo della parola in cui deve effettuare il trasferimento, 300 nel nostro

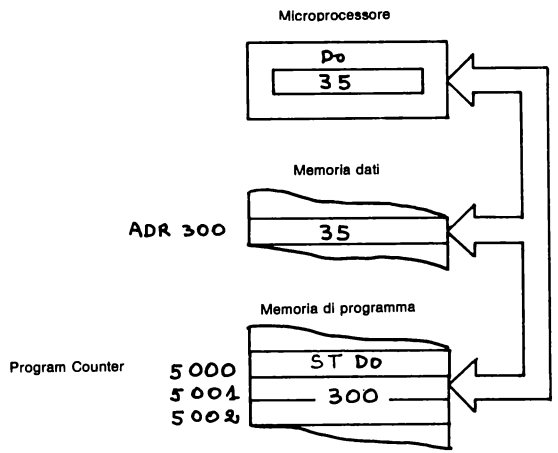


Figura 1 - *Indirizzamento esteso.*

caso (che corrisponde alla pagina 300 del libro). Poichè il contenuto del registro D0 è 35, il microprocessore carica 35 all'indirizzo 300. La figura 1 mostra il risultato finale di queste diverse operazioni.

INDIRIZZAMENTO INDIRETTO TRAMITE REGISTRI

Prendiamo l'esempio di un classificatore di documenti. Quelli che appartengono alla stessa rubrica sono raggruppati nello stesso capitolo. Due capitoli diversi vengono separati tra di loro da fogli forniti di indice, che specificano il nome della rubrica. L'accesso ai documenti di una stessa rubrica si fa cercando prima il talloncino del foglio indice di quella rubrica, e poi consultando i documenti in modo sequenziale. Il microprocessore, con l'indirizzamento indiretto tramite registri, effettua un'operazione analoga.

La figura 2 illustra questo principio. Un registro di indirizzo, per esempio A_2, è caricato con l'indirizzo della zona da consultare (corrispondente a un capitolo), ad esempio 300 nel caso della figura 2. Il registro D0 è caricato con il contenuto della locazione di memoria (35) puntata da A_2(indirizzo 300). L'istruzione eseguita è LD a A_2 D_0, che significa: carica indirettamente il registro D_0 con il contenuto della locazione di memoria puntata da A_2.

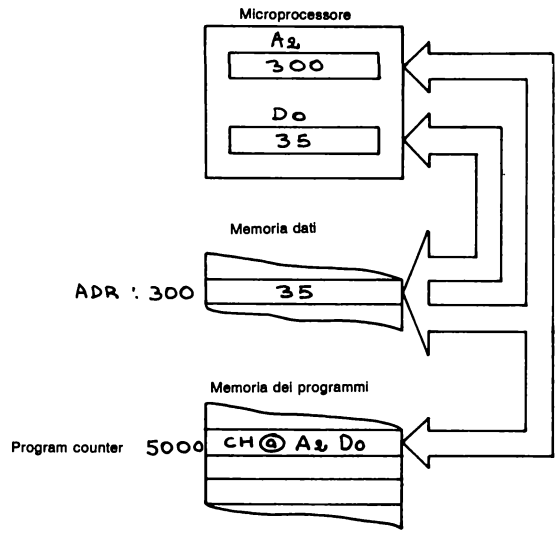


Figura 2 - *Indirizzamento indiretto tramite registri.*

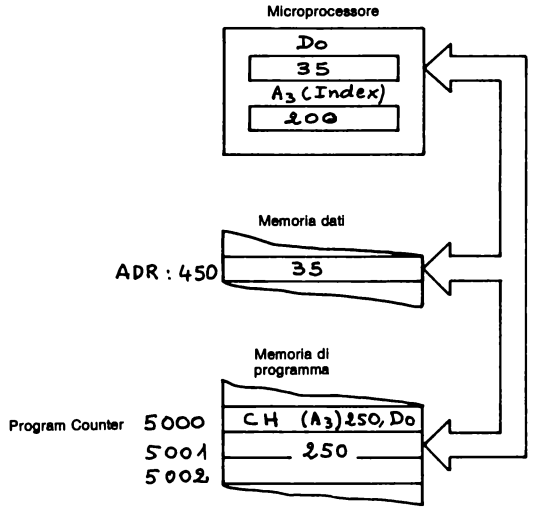


Figura 3 - *Indirizzo indicizzato.*

INDIRIZZAMENTO INDICIZZATO

Riprendiamo l'esempio del classificatore. Supponiamo di voler cercare nella rubrica "fornitori" quelli che vendono resistenze. Si arriva al talloncino "fornitori di resistenze", cercando prima la rubrica "fornitori", e, all'interno di questa, il capitolo riguardante i fornitori di resistenze.

Il microprocessore può eseguire la stessa operazione in due modi: tramite indirizzamento indicizzato semplice, o indirizzamento indiretto indicizzato.

Indirizzamento indicizzato semplice

L'indirizzo (450) dell'operando (35) è la somma dell'indirizzo specificato nell'istruzione (250) e dello spiazzamento (200) indicato nel registro indice (figura 3). Il microprocessore, eseguendo l'istruzione LD (A_3), 250, D_0, legge il contenuto della locazione di memoria di indirizzo (A_3) +250, cioè 450, e lo trasferisce in DO. Notiamo che l'indirizzo indicato nell'istruzione è l'analogo del capitolo "fornitori", e quello contenuto nel registro indice corrisponde al sotto-capitolo "fornitori di resistenze".

Indirizzamento indiretto indicizzato

L'indirizzo (450) dell'operando (35) è la somma del contenuto del registro di indirizzo A_2 e del contenuto del registro indice A_3 (figura 4).

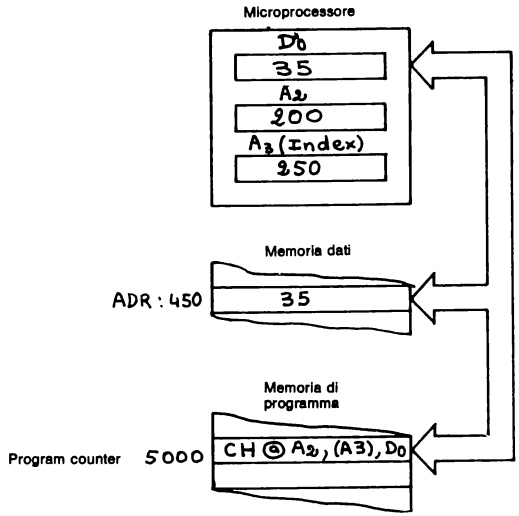


Figura 4 - *Indirizzamento indiretto indicizzato.*

Indirizzamento indiretto indicizzato con spiazzamento

Questo modo di indirizzamento è un caso particolare del precedente. L'indirizzo (470) dell'operando (35) è la somma del contenuto del registro di indirizzo A_2, del registro indice A_3 e dello spiazzamento indicato nell'istruzione (figura 5). Per

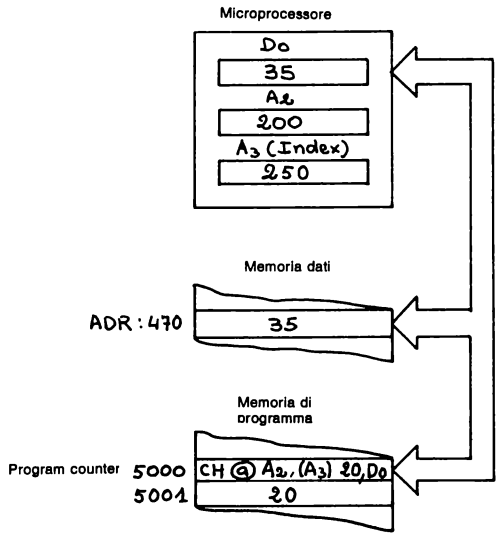


Figura 5 - *Indirizzamento indiretto indicizzato con spiazzamento.*

riprendere l'analogia con il classificatore, il contenuto del registro A_2 corrisponde al capitolo "fornitori", il contenuto di A_3 il sotto-capitolo "fornitori di resistenze", e lo spiazzamento il numero di un fornitore all'interno del "sotto-capitolo".

INDIRIZZAMENTO RELATIVO

I diversi modi di indirizzamento che abbiamo visto consentono di dare una struttura allo spazio di memoria e di accedere facilmente ai dati associati ad un programma. L'indirizzamento relativo aggiunge una certa flessibilità alla collocazione del o dei programmi in memoria, in quanto permette di creare programmi "rilocabili".

Un programma rilocabile è indipendente dalla sua posizione in memoria e non contiene indirizzi assoluti; può quindi essere eseguito in qualunque zona della memoria. Grazie alla rilocabilità, si possono registrare su ROM programmi che possono girare su macchine diverse, a indirizzi diversi. In questi programmi, le istruzioni di riferimento alla memoria usano l'indirizzamento relativo, con possibilità di salti relativi di almeno 32 kbyte in avanti e indietro. Poichè è necessaria una base di riferimento, la sequenza di inizializzazione del sistema usa l'indirizzamento assoluto, e contiene l'indirizzo di inizio del programma che viene caricato nel Program Counter.

Esistono due tipi di indirizzamento relativo: indirizzamento relativo lungo e indirizzamento relativo corto.

Indirizzamento relativo lungo

In questo indirizzamento si somma al contenuto del Program Counter lo spiazzamento con segno specifico nell'istruzione; dove "con segno" significa che lo spiazzamento può essere positivo o negativo (cioè in avanti o indietro rispetto all'indirizzo dell'istruzione).

Indirizzamento relativo corto

Permette di effettuare salti incondizionati o condizionati in uno spazio limitato di memoria (254 posizioni, nei microprocessori a 8 bit). Il bit più significativo indica il senso dello spiazzamento rispetto all'indirizzo attuale del Program Counter, e gli altri sette bit danno il valore assoluto dello spiazzamento (cioè + o — 127 posizioni). In questo modo di indirizzamento l'istruzione occupa due byte, uno per il codice operativo (ad esempio BR NZ = salta se diverso da zero), l'altro per lo spiazzamento con segno. Se questo è positivo, si possono saltare fino a 129 posizioni di memoria in avanti rispetto all'indirizzo dell'istruzione: in effetti, ai 127 passi che si possono rappresentare con i sette bit meno significativi, bisogna aggiungere, le due posizioni di memoria occupate dall'istruzione, in quanto l'indirizzo di salto è calcolato automaticamente a partire dall'indirizzo successivo all'i-

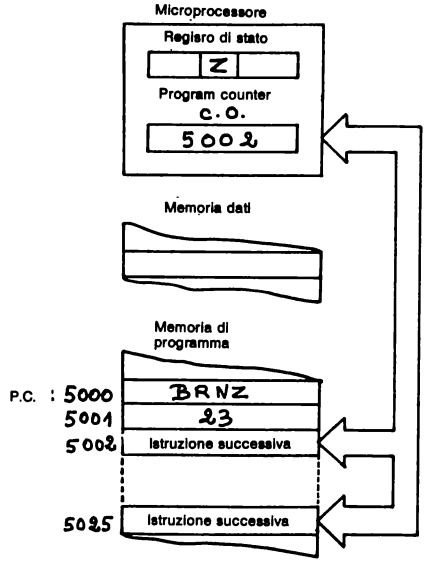


Figura 6 - *Indirizzamento relativo corto.*

struzione. Per esempio, se l'istruzione di salto si trova all'indirizzo 5000, sarà calcolato a partire da 5002 (cioè 5002 + 127 = 5129, con uno spiazzamento massimo di 129 passi, rispetto all'indirizzo dell'istruzione). Se lo spiazzamento è negativo, si può fare un salto di 125 passi indietro, rispetto all'indirizzo dell'istruzione. Riprendendo il nostro esempio, l'indirizzo finale di salto sarà 5002 - 127 = 4875, o 5000 — 125 = 4875.

La figura 6 mostra un esempio di salto condizionato con indirizzamento relativo corto. All'indirizzo 5000 della memoria di programma, il microprocessore trova

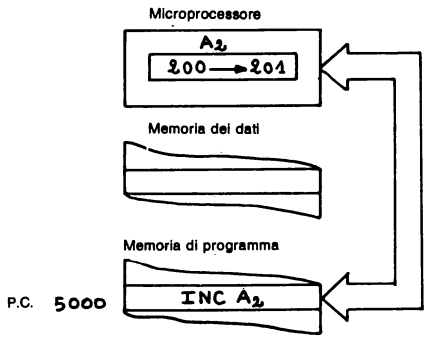


Figura 7 - *Indirizzamento implicito.*

l'istruzione BR NZ, seguita, all'indirizzo 5001, dallo spiazzamento (23 nel nostro caso). Se nel microprocessore, il registro dei flag segnala un risultato diverso da zero (in seguito a un'operazione precedente), si deve effettuare il salto. Il microprocessore aggiunge 23 a 5002 e salta all'indirizzo 5025, dove si troverà una nuova istruzione. Se il registro dei flag segnala un risultato nullo, il salto non viene effettuato, e il microprocessore prosegue dall'indirizzo corrente 5002.

Il modo di indirizzamento relativo corto ha il vantaggio di risparmiare memoria di programma e permette di effettuare facilmente salti, anche in programmi rilocabili.

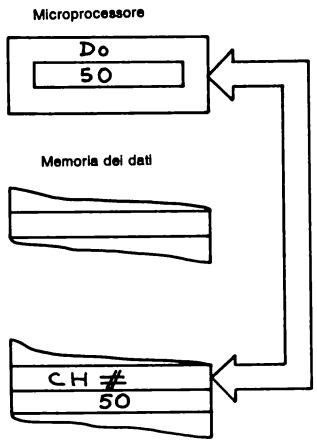


Figura 8 - *Indirizzamento immediato.*

INDIRIZZAMENTO IMPLICITO

Non si tratta di un vero e proprio modo di indirizzamento, ma permette di modificare il contenuto di un registro del microprocessore o di una parola di memoria. L'informazione necessaria per l'esecuzione dell'istruzione è contenuta nella stessa istruzione, e non ha bisogno di un indirizzo. La figura 7 mostra un esempio di indirizzamento implicito, che permette di incrementare il contenuto del registo A_2 ($INCA_2$).

INDIRIZZAMENTO IMMEDIATO

Non è vero e proprio modo di indirizzamento, perchè la parola che segue l'istruzione contiene un dato e non un indirizzo. Il dato in questo caso non è contenuto nella memoria dei dati ma nella memoria di programma. La figura 8 mostra l'indirizzamento immediato.

CAPITOLO 5

SET DI ISTRUZIONI

Il microprocessore è un circuito, la cui funzione definita da un programma che si trova in memoria, e che corrisponde a una sequenza di comandi che istruiscono il microprocessore sulle operazioni che deve eseguire. Queste istruzioni vengono definite una volta per tutte dalla casa costruttrice, e non possono essere modificate dall'utente, che le può solo usare per scrivere il suo programma.

L'insieme delle istruzioni di un microprocessore costituisce quello che si chiama il "set" di istruzioni, e comprende, in genere, sei gruppi:

1 — Istruzioni di trasferimento e di scambio;
2 — Istruzioni aritmetiche;
3 — Istruzioni logiche;
4 — Istruzioni di salto e ritorno al programma principale;
5 — Istruzioni di ingresso/uscita;
6 — Istruzioni di controllo.

In alcuni microprocessori esistono, inoltre, istruzioni di manipolazione dei bit, dei blocchi di dati, etc.

Nella figura 1 viene riportato l'esempio di un'organizzazione interna di un microprocessore, a cui abbiamo associato un set di istruzioni ben fornito (figure 2, 3, 4 e 5). Le istruzioni sono espresse con dei codici mnemonici, costituiti da un insieme di lettere che ricordano la loro funzione.

ISTRUZIONI DI TRASFERIMENTO E DI SCAMBIO

Comprendono (figura 2):

— le istruzioni (LD) di caricamento dei registri degli indirizzi o dei dati, in modo immediato o dalla memoria;
— le istruzioni (ST) di trasferimento in memoria dei contenuti dei registri;
— le istruzioni di "salvataggio" nell'area di stack dei contenuti dei registri (SAV);
— le istruzioni di ripristino dall'area di stack dei contenuti dei registri (RST);
— le istruzioni di scambio (EX) tra due registri.

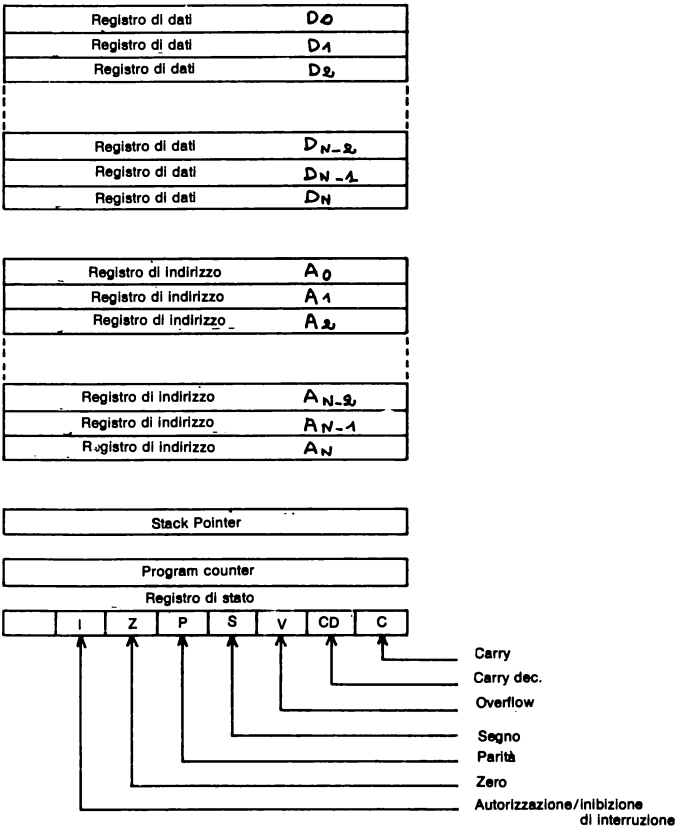
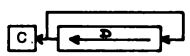
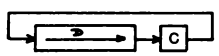
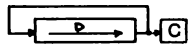
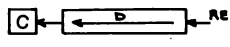
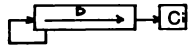
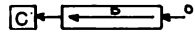
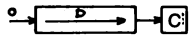
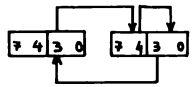
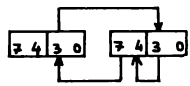


Figura 1 - *I registri del microprocessore accessibili dall'utente.*

Istruzioni	**Operazioni eseguite**	**Spiegazioni**
TRASFERIMENTI E SCAMBI		
LD	$(EA) \xrightarrow{(2)} R$ (1)	Caricamento di un registro con il contenuto di una locazione di memoria
EX	$Ri \rightleftarrows Rj$ (5)	Scambio dei contenuti tra due registri
RST	$MSP \longrightarrow R$ (4) $SP + N \longrightarrow SP$	Caricamento del registro R con il contenuto dell'area di stack puntata da SP. Incremento di SP della quantità N
SAV	$SP - N \longrightarrow SP$ $(R) \longrightarrow MSP$	Decremento di SP della quantità N. Salvataggio del contenuto di R nell'area di stack puntata da SP

Istruzioni	Operazioni eseguite	Spiegazioni
ST	$(R) \longrightarrow EA$	Trasferimento in memoria del contenuto di un registro
ARITMETICHE		
ADD	$(D) + (EA) \longrightarrow D$ (6)	Somma senza riporto (carry)
ADC	$(D) + (EA) + C \longrightarrow D$	Somma con riporto (carry)
ADCD	$(R)_{10} + (EA)_{10} + CD \longrightarrow D$	Addizione decimale con riporto decimale
SUB	$(D) - (EA) \longrightarrow D$	Sottrazione senza riporto
SBC	$(D) - (EA) - C \longrightarrow D$	Sottrazione con riporto
SBCD	$(D_{10}) - (EA)_{10} - CD \longrightarrow D$	Sottrazione decimale con riporto decimale
MULU	$(D) \times (EA) \longrightarrow DD$ (7)	Moltiplicazione senza segno
MULS	$(D) \times (EA) \longrightarrow DD$	Moltiplicazione con segno
DIVU	$(DD) / (EA) \longrightarrow DD$	Divisione senza segno. Risultato in DD
DIVS	$(DD) / (EA) \longrightarrow DD$	Divisione con segno. Risultato in DD
RES	$0 \longrightarrow EA$	Azzeramento
CPL	$0 - (EA) \longrightarrow EA$	Complemento a 2
CPLD	$0 - (EA)_{10} \longrightarrow EA$	Complemento a 10
INC	$(A) + N \longrightarrow A$	Incremento di N
DEC	$(A) - N \longrightarrow A$	Decremento di N
CMP	$(D) - (EA)$	Confronto
LOGIQUES		
AND	$(D) \wedge (EA) \longrightarrow D$	"AND" logico
OR	$(D) \vee (EA) \longrightarrow D$	"OR" logico
XOR	$(D) \oplus (EA) \longrightarrow D$	"EX-OR"
COM	$(\overline{D}) \longrightarrow D$	Complemento a 1
ROTATION ET DECALAGE		
RL		Rotazione a sinistra del registro D con carry
RLC		Rotazione a sinistra di D
RR		Rotazione a destra di D con carry

Istruzioni	Operazioni eseguite	Spiegazioni
RRC		Rotazione a destra di D
SLA		Shift aritmetico sinistro di D
SRA		Shift aritmetico destro di D
SLL		Shift logico sinistro di D
SRL		Shift logico destro di D
RRD		Rotazione di una cifra decimale a destra
RLD		Rotazione di una cifra decimale a sinistra

(1) EA	indirizzo effettivo, costruito con un modo di indirizzamento.
(2) (EA)	contenuto della parola di memoria puntata dall'indirizzo effettivo.
(3) R	registro di dati o di indirizzo.
(4) SP	stack pointer.
(5) MSP	contenuto della parola di memoria puntata da SP.
(6) D	registro dei dati.
(7) DD	doppio registro dei dati

Figura 2 - *Istruzioni di trasferimento e di scambio, aritmetiche, logiche, di rotazione e di shift.*

ISTRUZIONI ARITMETICHE

Operano su uno o due operandi, a seconda del tipo di istruzione (figura 2), e permettono di effettuare:

— le quattro operazioni fondamentali (addizione, sottrazione, moltiplicazione e divisione);
— l'azzeramento di un registro o di una locazione di memoria;
— il complemento a 2 o a 10 del contenuto di un registro o di una locazione di memoria;
— l'incremento o il decremento di una quantità N del contenuto di un registro o di una locazione di memoria;
— il confronto del contenuto di due registri, o di un registro e di una locazione di memoria.

La figura 6 mostra un esempio in cui vengono utilizzate le istruzioni di caricamento e le istruzioni aritmetiche.

ISTRUZIONI LOGICHE

Permettono di effettuare le quattro operazioni logiche fondamentali: AND, OR, OR-ESCLUSIVO, NOT.

ISTRUZIONI DI ROTAZIONE E DI SHIFT

Il microprocessore è in grado di eseguire shift e rotazioni a sinistra o a destra sul contenuto del registro dei dati D. Il numero di bit da shiftware è indicato nell'istruzione (figura 2). Due istruzioni speciali, RRD e RLD, permettono la rotazione di cifre decimali a sinistra e a destra (vedere esempi nelle figure 8, 9 e 10).

ISTRUZIONI A LIVELLO DI BIT

Permettono di indirizzare direttamente, all'interno di un registro o di una locazione di memoria, un bit, di verificarne il valore, di metterlo a zero o a uno (figura 3). Sono molto utili nel controllo dei processi industriali.

Istruzioni	Operazioni eseguite	Spiegazioni
A LIVELLO DI BIT		
BIT	$(\overline{EA})_{bit} \longrightarrow Z$	Selezione di un bit nella parola di memoria puntata da EA, complemento di questo bit in Z
RES	$0 \longrightarrow (EA)_{bit}$	0 in un bit della parola di memoria puntata da EA
SET	$1 \longrightarrow (EA)_{bit}$	1 in un bit della parola di memoria puntata da EA
BLOCCHI DI DATI		
LDD	SOURCE → DESTINATION	$M_{AS} \xrightarrow{(1)} M_{Ad}$; $A_S - 1 \longrightarrow A_S$; $A_d - 1 \longrightarrow A_d$; R_C (3) $- 1 \longrightarrow R_C$
LDDR	SOURCE → DESTINATION	$M_{AS} \longrightarrow M_{Ad}$; $A_S - 1 \longrightarrow A_S$, $A_d - 1 \longrightarrow A_d$; $R_C - 1 \longrightarrow R_C$ la sequenza viene ripetuta fino a che Rc = 0

Istruzioni	Operazioni eseguite	Spiegazioni
LDI	SOURCE→DESTINATION	$M_{AS} \rightarrow M_{Ad}$; $A_S+1 \rightarrow A_S$; $A_d+1 \rightarrow A_d$; $R_c-1 \rightarrow R_C$
LDIR	SOURCE→DESTINATION	$M_{AS} \rightarrow M_{Ad}$; $A_S+1 \rightarrow A_S$; $A_d+1 \rightarrow A_d$; $R_C-1 \rightarrow R_C$ la sequenza è ripetuta fino a che Rc = 0
CPD	D - DESTINATION	$D-M_{Ad}$; $A_d-1 \rightarrow A_d$; $R_C-1 \rightarrow R_C$; si $D-M_{Ad} = 0, Z = 1$
CPDR	D - DESTINATION	$D-M_{Ad}$; $A_d-1 \rightarrow A_d$, $R_C-1 \rightarrow R_C$, si $D-M_{Ad} = 0, Z = 1$ la sequenza è ripetuta fino a quando D-MAD = 0 o Rc = 0
CPI	D - DESTINATION	$D-M_{Ad}$; $A_d+1 \rightarrow A_d$; $R_C-1 \rightarrow R_C$; si $D - M_{Ad} = 0, Z = 1$
CPIR	D - DESTINATION	$D-M_{Ad}$; $A_d+1 \rightarrow A_d$; $R_C-1 \rightarrow R_C$; si $D - M_{Ad} = 0$ $Z = 1$ la sequenza è ripetuta fino a quando D-MAD = 0 o Rc = 0
CPSD	SOURCE-DESTINATION	$M_{AS}-M_{AD}$; $A_S-1 \rightarrow A_S$; $A_d-1 \rightarrow A_d$; $R_C-1 \rightarrow R_C$ si $M_{AS} - M_{Ad} = 0, Z = 1$
CPSDR	SOURCE-DESTINATION	$M_{AS}-M_{Ad}$; $A_S-1 \rightarrow A_S$; Ad-1→Ad ; $R_C-1 \rightarrow R_C$; si $M_{AS} - M_{Ad} = 0, Z = 1$ la sequenza è ripetuta fino a quando MAS—MAD = 0 o Rc = 0
CPSI	SOURCE-DESTINATION	$M_{AS}-M_{Ad}$, $A_S+1 \rightarrow A_S$; $A_d+1 \rightarrow A_d$; $R_C-1 \rightarrow R_C$ si $M_{AS} - M_{Ad} = 0, Z = 1$
CPSIR	SOURCE-DESTINATION	$M_{AS}-M_{Ad}$; $A_S+1 \rightarrow A_S$; $A_d+1 \rightarrow A_d$; $R_C-1 \rightarrow R_C$ si $M_{AS}-M_{Ad} = 0, Z = 1$ la sequenza è ripetuta fino a quando MAS—MAD = 0 o Rc = 0

(1) MAS parola di memoria della zona sorgente, puntata dal registro di indirizzo AS.

(2) M.AD parola di memoria della zona destinazione, puntata dal registro di indirizzo AD.

(3) Rc registro di conteggio.

Figura 3 - *Istruzioni a livello di bit e di manipolazione dei caratteri.*

ISTRUZIONI DI MANIPOLAZIONE DI CARATTERI

Sono descritte nella figura 3, e comrpendono le istruzioni di ricopiatura, di ricerca di un carattere all'interno di un blocco di dati, e di confronto tra due blocchi di dati, carattere per carattere.

Le istruzioni di ricopiatura trasferiscono il contenuto di una locazione di memoria M_{As}, puntato al registro di indirizzo A_s, in una locazione di memoria MA_d puntata dal registro di indirizzo A_d, incrementando o decrementando A_s e A_d, e decrementando il registro di conteggio R_C, che all'inizio contiene la lunghezza in caratteri della zona da ricopiare. Alcune istruzioni, inoltre, se il contenuto di R_C è diverso da zero, effettuano un trasferimento della nuova locazione di memoria puntata da A_s nella nuova locazione di memoria puntata da A_d, e così di seguito fino a quando il contenuto di R_C è uguale a zero. Un esempio di programma di ricopiatura è mostrato nelle figure 11 e 12.

Le istruzioni di ricerca di un carattere all'interno di un blocco di dati confrontano il contenuto di un registro di dati D con quello della posizione di memoria M_{Ad}, puntata dal registro di indirizzo A_d, incrementano o decrementano A_d e decrementano il registro di conteggio R_C, che contiene la lunghezza del blocco di dati in caratteri. Alcune istruzioni, se non c'è identità tra i due contenuti, o se il contenuto di R_C è diverso da zero, effettuano un nuovo confronto tra il contenuto di D e la nuova locazione di memoria puntata da A_d. L'operazione di ricerca del carattere, in questo caso, si arresta quando c'è identità tra il contenuto di D e quello della locazione di memoria puntata da A_d, o quando il contenuto di R_C è uguale a zero. Un esempio di ricerca di caratteri è mostrato nelle figure 13 e 14.

Le istruzioni di confronto di due blocchi, carattere per carattere, assomigliano alle istruzioni precedenti. Permettono di confrontare il contenuto di memoria del blocco sorgente con quella corrispondente del blocco destinazione.

SALTI E RITORNI AL PROGRAMMA PRINCIPALE

I salti possono essere incondizionati o condizionati. I salti condizionati dipendono dal contenuto di un registro di stato, che contiene un certo numero di flip flop che si posizionano a zero o a uno, a seconda del risultato dell'esecuzione di un'istruzione: risultato positivo o negativo, risultato nullo o diverso da zero, riporto, overflow, etc. A seconda dell'informazione memorizzata in questi "flag", l'istruzione di salto condizionato sarà eseguita o meno. Le istruzioni di salto elencate nella figura 4 dipendono da una condizioni o da un'operazione logica su due o tre condizioni.

ISTRUZIONI DI INGRESSO USCITA

Poichè i registri delle interfacce sono considerati come locazioni di memoria, le istruzioni di ingresso/uscita sono analoghe a quelle di trasferimento. (In alcuni microprocessori i registri di ingresso/uscita non vengono trattati come locazione di memoria. In quel caso esistono istruzioni particolari di ingresso/uscita).

Istruzioni	Flag modificati	Spiegazioni
BR	Z = 1	Salto incondizionato
BRZ	Z = 0	Salto se = 0
BRNZ	C = 1	Salto se ≠ 0
BRC	C = 0	Salto se C = 1
BRNC	V = 1	Salto se C = 0
BRV	V = 0	Salto se V = 1
BRNV	S = 1	Salto se V = 0
BRS	S = 0	Salto se negativo
BRNS	C⊕Z = 1	Salto se positivo
BRIE	S⊕V = 1	Salto se minore o uguale
BRIZ	(S + V) + Z = 1	Salto se minore di zero
BRIEZ	C + Z = 1	Salto se minore o uguale a zero
BRS	(S⊕V) + Z = 0	Salto se maggiore
BRSZ	S⊕V = 0	Salto se maggiore di zero
BSEZ	P = 1	Salto se maggiore o uguale a zero
BRP	P = 0	Salto se parità pari
BRNP		Salto se parità dispari
DBNZ		Decremento del contatore e salto se ≠ da zero
CALL		Chiamata a sottoprogramma
RET		Ritorno al programma principale dopo l'esecuzione di un sottoprogramma
RETI		Ritorno al programma principale dopo il servizio di un'interruzione

Figura 4 - *Istruzioni di salto condizionato o incondizionato e di ritorno al programma principale.*

ISTRUZIONI DI CONTORLLO DELL'UNITA' CENTRALE

Queste istruzioni (figura 5) permettono di posizionare a zero o a uno i flag del registro di stato, di abilitare o disabilitare le interruzioni e di fermare il microprocessore.

Istruzioni	Spiegazioni
RESZ	Z = 0
SETZ	Z = 1
RESC	C = 0 (carry)
SETC	C = 1 (carry)
RECD	CD = 0 (carry decimale)
SECD	CD = 1 (carry decimale)
RESV	V = 0 (overflow)
SETV	V = 1 (overflow)
RESS	S = 0 (segno)
SETS	S = 1 (segno)
RESP	P = 0 (parità)
SETP	P = 1 (parità)
EI	Abilitazione delle interruzioni
DI	Disabilitazione delle interruzioni
NOT	Nessuna operazione
HALT	Arresto del microprocessore

Figura 5 - *Istruzioni di controllo dell'unità centrale.*

ESEMPI DI PROGRAMMAZIONE

Concludiamo questo capitolo con qualche esempio di programmazione.

Addizione decimale di due numeri di tre cifre

Questo programma permette di sommare due numeri decimali di tre cifre, memorizzati rispettivamente agli indirizzi 100, 101 e 102 il primo, e 200, 201 e 202 il

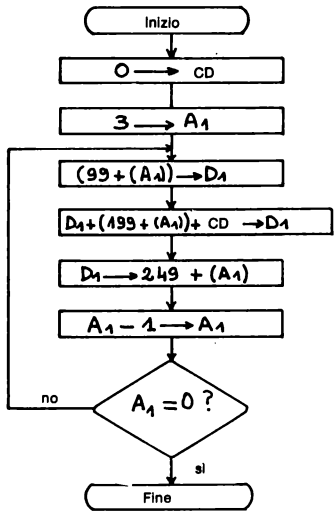


Figura 6 - *Diagramma a blocchi di un programma di somma.*

Istruzioni	Spiegazioni
RECD	Azzeramento del flag di carry decimale CD;
LD A1, ≠ 3	Caricamento del numero 3 nel registro di indirizzo A1;
LOOP LD(A1)99,D1	Caricamento del registro di dati D1 con il contenuto della parola di memoria il cui indirizzo è 99 + A1;
ADCD D1, (A1)199	Addizione decimale con riporto del contenuto di D1 e della parola di memoria di indirizzo 199 + A1;
ST D1, (A1)249	Memorizzazione del risultato nella parola di memoria di indirizzo 249 + A1;
DBNZ A1, LOOP	Decremento di un'unità di A1, e salto a LOOP se A1 è diverso da zero;
HALT	Arresto del microprocessore

Figura 7 - *Programma di addizione di due numeri decimali a tre cifre.*

secondo. Il risultato è memorizzato negli indirizzi 250, 251 e 252 (supponiamo che l'addizione porti a un risultato di tre cifre).

Il diagramma a blocchi che rappresenta le diverse fasi del programma è descritto nella figura 6. Il programma è descritto nella figura 7.

Shift di una cifra a destra (microprocessori a 8 bit)

Questo esempio mostra come shiftare un numero di sei cifre a destra. Il diagramma a blocchi è descritto nella figura 8, e la figura 9 mostra che la posizione 200 è la prima di una zona di memoria contenente 6 cifre (9,5,7,3,2,1), collocate agli indirizzi 200, 201 e 202.

La figura 10 mostra il programma che effettua lo shift.

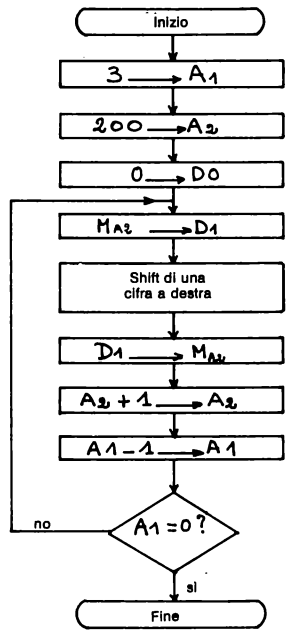


Figura 8 - *Diagramma a blocchi dell'operazione di shift a destra di una cifra.*

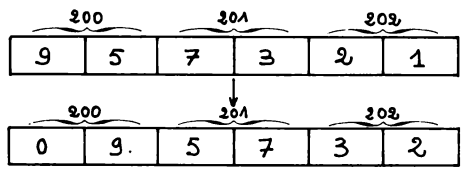


Figura 9 - *Principio di funzionamento dello shift di una cifra a destra.*

Programma	Spiegazioni
LD A1, ≠ 3	Caricamento di A1 con 3;
LD A2, ≠ 200	Caricamento di A2 con 200;
RES D0	Azzeramento di D0;
LOOP LD ⓐ A2, D1	Caricamento di D1 con il contenuto della parola di memoria puntata da A2;
RRD D0, D1	Shift di una cifra a destra;
LD D1, ⓐ A2	Caricamento della parola di memoria puntata da A2 con il contenuto di D1;
INC A2	Incremento di 1 di A2;
DBNZ A1, LOOP	Decremento di 1 di A1 e salto a LOOP se A1 è diverso da zero;
HALT	Arresto del microprocessore.

Figura 10 - *Programma di shift di una cifra a destra.*

Trasferimento di un blocco di dati (microprocessore a 8 bit)

Questo programma permette di trasferire una zona di 512 byte. La zona sorgente inizia all'indirizzo ZNSRCE, e la zona destinazione all'indirizzo ZNDST. Il diagramma a blocchi è mostrato nella figura 11, e il programma nella figura 12.

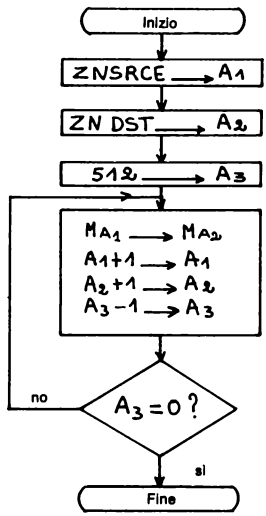


Figura 11 - *Diagramma a blocchi di un trasferimento.*

Programma	Spiegazioni
LD A1, ≠ ZNSRCE	Caricamento di A1 con l'indirizzo di inizio della zona sorgente;
LD A2, ≠ ZNDST	Caricamento di A2 con l'indirizzo di inizio della zona destinazione;
LD A3, ≠ 512	Caricamento di A3 con la lunghezza della zona sorgente in byte;
LDIR ⓐ A1, ⓐ A2, A3	Ricopiatura di 512 byte;
HALT	Arresto del microprocessore.

Figura 12 - *Programma di trasferimento di un blocco di dati.*

Ricerca di un carattere in un blocco di dati (microprocessore a 8 bit)

Questo programma ricerca un carattere di ritorno (%R) all'interno di un blocco di caratteri di 512 byte. Il diagrmma a blocchi è descritto nella figura 13 e il programma nella figura 14.

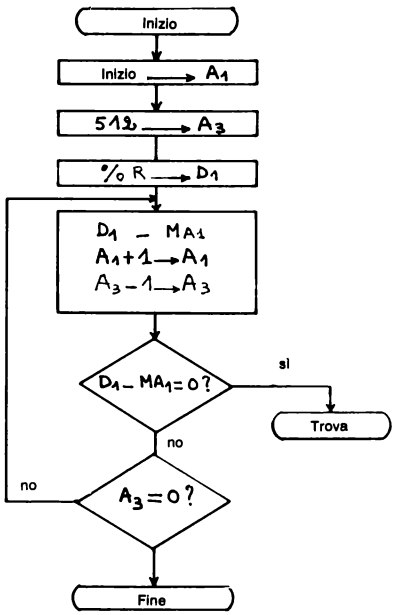


Figura 13 - *Diagramma a blocchi di una ricerca di caratteri.*

Programma	Spiegazioni
LD A1, ≠ INIZIO	Caricamento di A1 con l'indirizzo di inizio del blocco di caratteri;
LD A3, ≠ 512	Caricamento di A3 con la lunghezza in byte della zona di memoria;
LD D1, ≠ "%R"	Caricamento di D1 con %R;
CPIR ⓐA1, D1, A3, EQ	Ricerca del carattere;
BRZ, TROVA	Salta a "TROVA" se il carattere è trovato.

Figura 14 - *Programma di ricerca di un carattere di un blocco di dati.*

CAPITOLO 6

I SISTEMI DI SVILUPPO

Il microprocessore comprende un solo linguaggio, quello binario, in cui sia le istruzioni che gli indirizzi e i dati sono espressi da una sequenza di 1 e di 0. Nei microprocessori con la parola a 8 bit, un'istruzione si rappresenta con 8 bit, cioè con una sequenza di otto 0 e 1. Ad esempio, l'istruzione di caricamento di una locazione di memoria nel registro D_1 si esprime con: 10110110.

La programmazione di un'istruzione di questo genere comporta la manipolazione di otto interruttori, che saranno alti o bassi a seconda che il bit corrispondente sia uno o zero.

LINGUAGGIO ESADECIMALE

La codifica in binario (o linguaggio macchina) è naturalmente molto pesante. Per semplificarla, si è pensato di usare il linguaggio esadecimale, che è più compatto del binario. Si è visto che con quattro bit è possibile contare fino a quindici (figura 1).

Per non avere a che fare con numeri a due cifre, i numeri da 10 a 15 sono sostituiti dalle lettere da A a F. Nei microprocessori a 8 bit, una istruzione si esprime con due simboli: due cifre, una cifra e una lettera o due lettere. Nell'esempio precedente, il

Linguaggio binario	Linguaggio esadecimale
0 0 0 0	0
0 0 0 1	1
0 0 1 0	2
0 0 1 1	3
0 1 0 0	4
0 1 0 1	5
0 1 1 0	6
0 1 1 1	7
1 0 0 0	8
1 0 0 1	9
1 0 1 0	A
1 0 1 1	B
1 1 0 0	C
1 1 0 1	D
1 1 1 0	E
1 1 1 1	F

Figura 1 - *Corrispondenza tra il linguaggio binario e quello esadecimale.*

caricamento di una locazione di memoria in D_1 si esprimerebbe con B6. Invece di manipolare otto interruttori basta premere due tasti di una tastiera esadecimale. ma poichè il microprocessore capisce solo il linguaggio binario, è necessario un "traduttore" tra l'uomo che programma in linguaggio esadecimale e il microprocessore (figura 2). Questo traduttore è un piccolo programma, che si chiama monitor, che permette di inserire da tastiera un programma esadecimale, e ne assicura la conversione da esadecimale in binario.

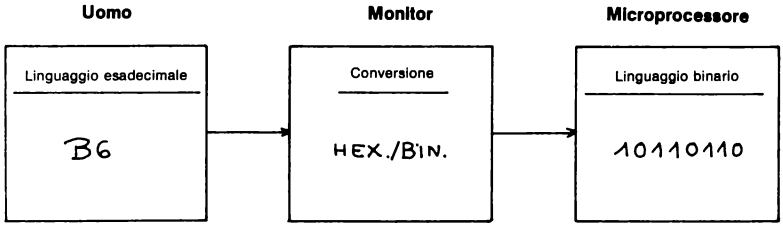


Figura 2 - *Programmazione in esadecimale (caricamento del registro D_1 con il modo di indirizzamento esteso).*

Un microcalcolatore, fornito di una tastiera esadecimale, di alcuni display, del microprocessore e di qualche RAM, contiene, in genere, un monitor, memorizzato in ROM, che permette, oltre alla conversione esadecimale/binario, di effettuare le seguenti operazioni:

— visualizzare e eventualmente modificare i contenuti dei registri del microprocessore;
— visualizzare e eventualmente modificare i contenuti delle locazioni di memoria del microcalcolatore;
— registrare i programmi su cassette magnetiche;
— caricare in RAM programmi da cassette magnetiche;
— lanciare un programma memorizzato nella RAM;
— eseguire un programma, un'istruzione alla volta (single step), etc.

LINGUAGGIO ASSEMBLER

Il codice esadecimale di un'istruzione non richiama alla mente la funzione che quella istruzione compie, e questo è un grosso inconveniente. Per programmi più lunghi di un centinaio di parole, è conveniente per il programmatrore usare un linguaggio più comprensibile. Si usa allora il linguaggio assembler, che è costituito da istruzioni di più lettere, scelte in modo da suggerire all'utente la funzione dell'istruzione stessa. Così l'istruzione LDD$_1$ (LOAD D1) specifica un caricamento di D_1 col modo di indirizzamento esteso, e l'informazione che segue è l'indirizzo della locazione di memorioa da caricare in D_1.

Il microprocessore non capisce nè il linguaggio esadecimale, nè quello assembler. Qualsiasi programma scritto in linguaggio assembler (programma sorgente) deve essere tradotto in binario; il risultato della traduzione si chiama "programma oggetto" (figura 3).

La traduzione viene eseguita da un programma speciale, detto assemblatore, che, rispetto al monitor esadecimale, è molto più complesso. In effetti, oltre alla traduzione l'assemblatore segnala eventuali errori. Per esempio, se invece di battere alla tastiera l'istruzione LDD1, il programmatore scrive LD1, l'assemblatore segnala, in fase di assemblaggio, che l'istruzione LD1 non esiste nel suo set di istruzioni.

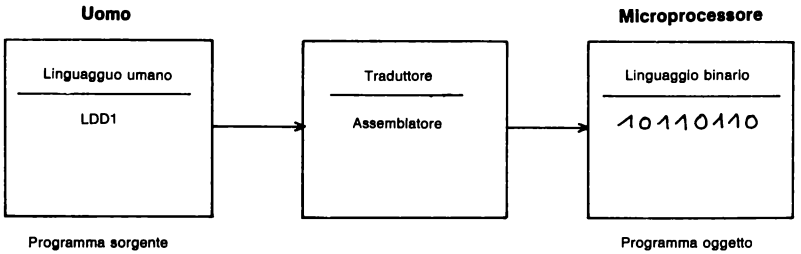


Figura 3 - *Programmazione in assembler.*

Un assemblatore è utilizzato, in genere, su un sistema più completo di una scheda a microprocessore, che si chiama sistema di sviluppo.

Si tratta in realtà di un microprocessore, collegato in genere a una tastiera-video, una stampante e un'unità a floppy-disk. Dispone, oltre al programma assemblatore, di altri programmi che consentono l'"editing", il caricamento, la messa a punto, etc.

Il programma di "editing", o editor, permette di inserire dalla console un programma sorgente (scritto in linguaggio assembler) e di trasferirlo su di un floppy-disk. Con questo programma l'utente può facilmente aggiungere, eliminare o modificare porzioni del programma sorgente memorizzato su un floppy disk.

Il programma di caricamento, o loader, permette di trasferire un programma oggetto dal floppy-disk alla memoria centrale del microprocessore. Esistono due tipi di programmi di caricamento: alcuni caricano il programma oggetto a partire da un indirizzo fisso, altri permettono di caricarlo in diverse zone della memoria, e prendono il nome di loader rilocabili.

Il programma di messa a punto, detto anche monitor, permette come dice il suo nome, di mettere a punto un programma oggetto. Fornisce le seguenti funzioni:

— visualizzazione ed eventuale modifica del contenuto di un registro del microprocessore o di una locazione di memoria;
— esecuzione di un programma oggetto, memorizzato nella memoria centrale del sistema di sviluppo;

— arresto dell'esecuzione di un programma in certi punti (indirizzi di memoria, condizioni indicate dall'utente), e visualizzazione dei registri del microprocessore;
— esecuzione istruzioni per istruzione (single step) e visualizzazione dei registri interni del microprocessore; etc.

Il sistema di sviluppo è in grado di verificare un programma in tempo reale, cioè di pilotare una qualsiasi applicazione con un programma che si trova nella memoria centrale del sistema di sviluppo stesso. Per far questo, una specie di "cordone ombelicale" collega l'applicazione al sistema di sviluppo (figura 4).

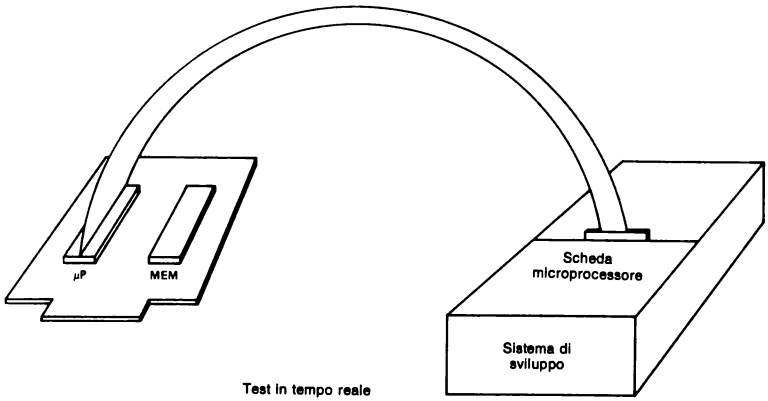


Figura 4 - *Test in tempo reale di un programma.*

Dalla parte dell'applicazione, il cordone è collegato al supporto del microprocessore (per il quale è prevista la collocazione sulla scheda dell'applicazione); dalla parte del sistema è direttamente collegato al microprocessore. Si può anche lanciare l'esecuzione del programma, a partire dal sistema di sviluppo, e verificare che al livello dell'applicazione il programma esegue le funzioni previste. Se non è così, si cerca, con l'aiuto del programma di messa a punto, la causa del cattivo funzionamento; si modifica il programma sorgente con l'editor, e si procede al riassemblaggio. Questa operazione di messa a punto viene ripetuta fino a quando l'applicazione viene gestita correttamente dal sistema di sviluppo. A questo punto si trasferisce il programma oggetto dalla memoria centrale su delle memorie EPROM (questa fase di programmazione delle EPROM è anch'essa realizzata dal sistema di sviluppo). Le EPROM sono poi inserite nei supporti a loro riservati sulla scheda dell'applicazione. Si procede quindi ad un'ulteriore prova, a partire dalle EPROM dell'applicazione, con il microprocessore del sistema di sviluppo. Si verifica infine che tutto funzioni correttamente, e se è il caso, si sconnette il cordone ombelicale per mettere al suo posto il microprocessore. Questa possibilità di emulazione in tempo reale,

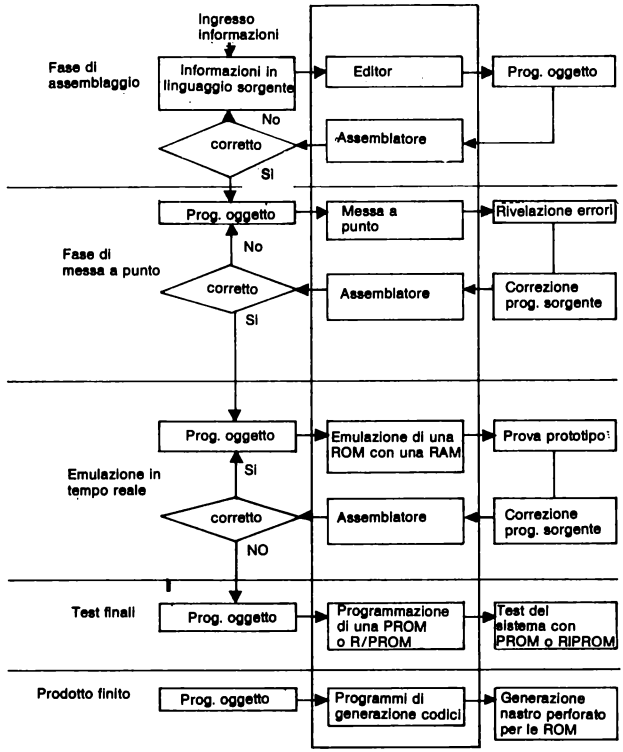


Figura 5 - *Sviluppo del software di un sistema basato su di un microprocessore.*

che abbiamo appena descritto, è molto importante, e permette di riunire, in un solo ciclo di sviluppo, le diversi fasi di studio del software e dell'hardware.

La figura 5 riporta in forma schematica le diverse fasi di sviluppo di un'applicazione.

LINGUAGGI EVOLUTI

Un sistema di sviluppo permette anche di programmare in linguaggio evoluto: BASIC, FORTRAN, COBOL, PL1, PASCAL. A differenza del linguaggio Assembler, che è molto pesante da manipolare, i linguaggi evoluti presentano il vantaggio della semplicità. Le istruzioni si esprimono in forma chiara, facilmente accessibile all'uomo, perchè sono costituite da comuni vocaboli della lingua inglese. Inoltre un linguaggio evoluto è un linguaggio standard, costruito a partire da certe specifiche. Un programma scritto in BASIC, per esempio, può girare, eventualmente con qualche piccola modifica, su divesi sistemi di sviluppo, destinati a diversi tipi di

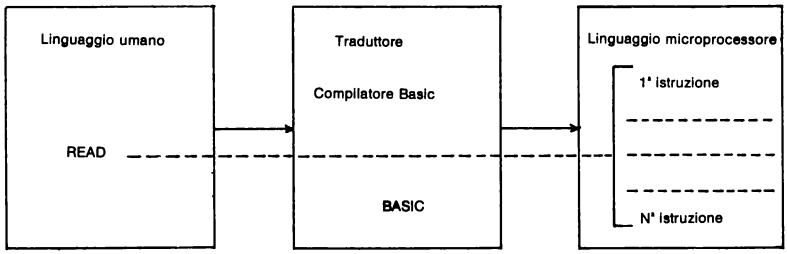


Figura 6 - *Linguaggio evoluto.*

microprocessori. Cambiano soltanto i compilatori, che traducono il programma scritto in linguaggio evoluto, in linguaggio binario, e che fanno parte del software di base di una certa macchina (figura 6).

Un programma scritto in linguaggio evoluto, essendo più vicino all'uomo, non sfrutta in modo ottimale tutte le possibilità del microprocessore, e occupa più spazio in memoria dello stesso programma scritto in assembler. Ha senso quindi sviluppare un'applicazione in linguaggio evoluto se deve essere prodotta in piccola serie, perchè, in questo caso, i costi dello sviluppo del software hanno un grosso peso sui costi di produzione. Viceversa applicazioni di grande serie saranno progettate in linguaggi assembler.

I linguaggi evoluti offrono molti altri vantaggi, oltre alla programmazione di applicazioni in piccola serie. Permettono di verificare, prima di programmare in assembler, se un'applicazione in grande serie è realizzabile: si programma rapidamente l'applicazione in linguaggio evoluto, e si verifica se l'approccio adottato porta o no ai risultati desiderati. Inoltre un linguaggio evoluto permette di utilizzare il sistema di sviluppo come un piccolo elaboratore per applicazioni connesse a quella in corso.

```
100  READ A, B, C, D
110  LET R = A * D - B * C
120  IF R = 0 THEN 180
130  READ P, Q
140  LET X = (P * D - B * Q)/R
150  LET Y = (A * Q - P * C)/R
160  PRINT X, Y
170  GO TO 130
180  PRINT "PAS DE SOLUTION UNIQUE"
190  DATA 5, 4, 4, 6
200  DATA 22, 26
210  DATA  7,  0
999  END
```

Dopo l'esecuzione del programma il microcalcolatore stamperà:

```
 2  3
 3 -2
```

Figura 7 -

Vediamo ora un esempio di programmazione in BASIC, che consiste nel risolvere un sistema di due equazioni di primo grado.

$$ax + by = p$$
$$cx + dy = q$$

Questo sistema ammette soluzione se la quantità ad — bc è diversa da zero. In questo caso si ottengono i seguenti risultati:

$$x = \frac{pd - bq}{ad - bc} \qquad y = \frac{aq - pc}{ad - bc}$$

Quando la quantità ad — bc è uguale a zero, non esiste soluzione. Il programma che permette di trattare questo sistema è mostrato nella figura 7.

Esaminandolo si possono fare le seguenti osservazioni: sono usate solo lettere maiuscole; ogni linea del programma comincia con un numero di linea, che caratterizza questa linea e determina l'ordine in cui devono essere eseguite le istruzioni. Ogni linea contiene, oltre al numero di linea, un'istruzione e un operando. Questa istruzione è generalmente in inglese. Gli spazi tra il numero di linea e l'istruzione, da una parte, e l'istruzione e l'operando, dall'altra non sono necessari, se non per la stampa di un messaggio. La loro funzione è di rendere più chiaro il programma e di facilitarne la lettura.

Torniamo al programma, analizzando linea per linea. La linea 100 contiene l'istruzione READ (LEGGERE), che è una parola dell'inglese corrente. La seguono quattro variabili, A, B, C, D. L'istruzione READ comporta la presenza nel programma di una o più istruzioni DATA (DATI), che permettono di attribuire ad ogni variabile uno o più valori, a seconda che il programma sia ripetuto una o più volte.

Nel nostro esempio si assegna alla variabile A il valore 5, fornito dall'istruzione 190. Analogamente si attribuiscono a B, C, D, i valori 4, 4, 6 rispettivamente. L'istuzione (LET) che si trova nella linea 110, fa eseguire al microelaboratore l'operazione indicata a destra del segno =, cioè A x D — B x C. Notiamo che in BASIC non esiste il segno di moltiplicazione x, ed è sostituito dal simbolo x, che deve essere scritto ogni volta che c'è da calcolare una moltiplicazione. La macchina non interpreterà AD come la moltiplicazione di A per D. L'espressione corretta è A x D. La linea 120 è un salto condizionato espresso con vocaboli dell'inglese corrente. Controlla il risultato R, e, se R è uguale a zero (IF R = 0), allora (THEN) salta alla linea 180. La linea 180 contiene l'ordine di stampa (PRINT) del messaggio "NON ESISTE UNA SOLUZIONE UNICA". Dopo la linea 180, il microelaboratore dovrebbe eseguire le linee 190, 200 e 210. Queste linee contengono dati, e non comportano l'esecuzione di un'istruzione. Di conseguenza il microelaboratore passa alla linea 999 e termina il programma. Se il risultato R è diverso da zero, il microelaboratore, dopo avere eseguito l'istruzione della linea 120, passa all'istruzione della linea 130 che richiede un'operazione di lettura (READ). L'esecuzione

della READ attribuisce ai parametri P e Q i valori corrispondenti indicati nella linea 200 (cioè 22 e 26). Le istruzioni delle linee 140 e 150 fanno calcolare al microelaboratore le variabili X e Y.

Notiamo le parentesi, che sono necessarie perchè vengono divise per R le quantità P x D — B x Q o A x Q — P x C. Se le parentesi fossero omesse, la macchina dividerebbe le quantità B x Q e P x C per R.

Notiamo anche che non esiste il segno : che potrebbe essere confuso con i due punti del linguaggio corrente. Questo segno è sostituito dal simbolo /. L'istruzione della linea 160 fa stampare al microprocessore le variabili X e Y. Poi, seguendo l'istruzione di salto GO TO (vai a) della linea 170, il microcalcolatore torna alla linea 130, assegna ai parametri P e Q i valori corrispondenti della linea 210, e calcola di nuovo X e Y. Dopodichè torna alla linea 130 e così di seguito fin quando non trova più dati. Allora termina il programma...

CAPITOLO 7

DESCRIZIONE DEI PRINCIPALI MICROPROCESSORI ESISTENTI SUL MERCATO

In quest'ultimo capitolo si descrivono i principali microprocessori esistenti sul mercato e, in particolare, la loro organizzazione esterna e interna, il set delle istruzioni e le interfaccie.

Elenco dei microprocessori descritti:

— Motorola MC 6800
— Motorola MC 6809
— MCS 6502 a tecnologia MOS
— Intel 8080 A
— Zilog Z 80
— Signetics 2650
— RCA CDP 1802
— SC/MP II della National Semiconductor
— Fairchild F8
— Intersil IM 6100
— Texas TMS 9900
— Intel 8086
— Zilog Z 8000
— Motorola MC 68000
— National Semiconductor NS 16000

L'autore ringrazia la rivista "Minis et Micros" per l'autorizzazione a riprendere un certo numero di illustrazioni comparse in questa pubblicazione.

MOTOROLA MC 6800

Il microprocessore MC 6800 è di uso generale. I bus dei dati e degli indirizzi sono separati. La capacità di indirizzamento è di 65.536 byte. I registri di ingresso/uscita delle interfaccie sono trattati come locazioni di memoria. Ha 7 modi di indirizzamento: tramite registri, in pagina zero, esteso, indicizzato, relativo, immediato e implicato. Ha 6 registri programmabili dall'utente: due accumulatori di 8 bit, un registro indice di 16 bit, uno stack-pointer di 16 bit, un program counter di 16 bit e un registro di stato di 8 bit. La configurazione minima richiede 5 chip: il clock, il microprocessore, una RAM, una ROM e un'interfaccia. Esistono tre versioni del 6800, che operano rispettivamente a 1 MHz, 1,5 MHz e 2 MHz.

Set di istruzioni

Possiede 72 tipi di istruzioni. Tenendo conto dei modi di indirizzamento, arriva a 197 istruzioni, che possono essere classificate in cinque gruppi: istruzioni di trasferimento, istruzioni aritmetiche, istruzioni logiche, istruzioni di salto incondizionato e condizionato e di chiamata ai sottoprogrammi, istruzioni di controllo del microprocessore. La lunghezza delle istruzioni è di 1, 2 o 3 byte. La maggior parte delle istruzioni operano sull'accumulatore e su un operando.

Interfaccie

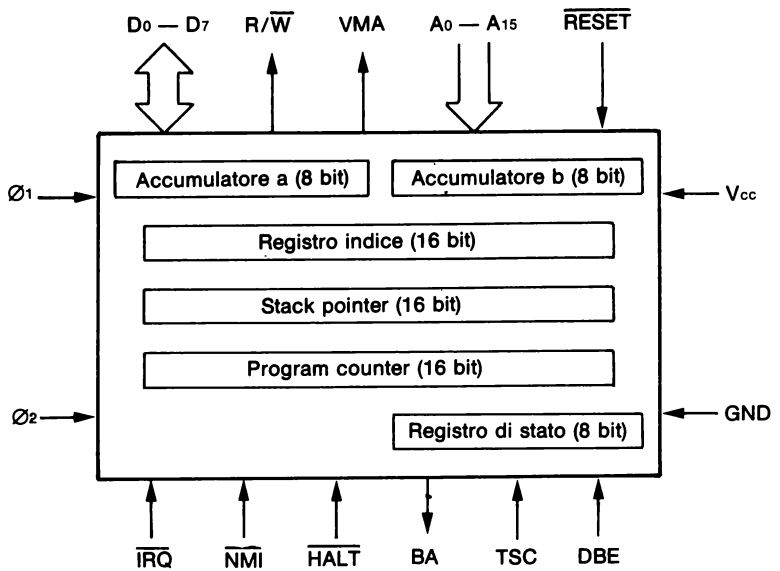
— interfaccia parallela (PIA): MC 6820; 16 linee di dati programmabili individualmente; 4 linee di controllo di cui due programmabili come ingresso o come uscita;
— interfaccia serie asincrona (ACIA): MC 6850;
— interfaccia serie sincrona (SSDA); MC 6852;
— interfaccia che opera secondo i protocolli HDLC/SDLC: MC 6854;
— interfaccia video MC 6845 o EF 9364 (EFCIS);
— interfaccia per unità a floppy-disk MC 6843;
— interfaccia per unità a cassetta megnetica: HD 46502 A 02 (HITACHI);
— accesso diretto alla memoria (DMA) MC 6844;
— temporizzatore/contatore MC 6840;
— interfaccia per strumentazione MC 68488.

Seconde sorgenti

EFCIS, Ami, Hitachi, Fujitsu e Fairchild.

MC 6800

Organizzazione esterna e interna



Nome dei piedini	Significato
D_0 - D_7	Bus dati
R/$\overline{W}$	Selezione lettura/scrittura o ingresso/uscita
VMA	Indirizzo di memoria valido
A_0 - A_{15}	Bus di indirizzo
$\overline{RESET}$	Ritorno allo stato iniziale
Vcc	Alimentazione: + 5 V
GND	Massa
DBE	Abilitazione Bus Dati
TSC	Posizionamento nello stato di alta impedenza
BA	Riconoscimento di richiesta DMA
$\overline{HALT}$	Richiesta DMA
$\overline{NMI}$	Richiesta di interruzione non mascherabile
$\overline{IRQ}$	Richiesta di interruzione mascherabile
$Ø_1$, $Ø_2$	Impulsi di clock

MOTOROLA MC 6809

Il Motorola MC 6809 è una versione più potente del 6800. Compatibile a livello di linguaggio sorgente con il 6800 e il 6802 ha, in più di questi, istruzioni che operano su 16 bit e modi di indirizzamento più efficienti. La sua architettura, prevista per lavorare sia su 8 che su 16 bit, è strutturata intorno a quattro registri a 8 bit, e cinque a 16 bit. I primi sono gli accumulatori A e B, il registro dei flag e il registro di pagina. I due accumulatori possono essere riuniti per formarne uno solo (D) a 16 bit. Gli altri quattro registri sono destinati all'indirizzamento della memoria: registri indice X e Y; registro di indirizzamento indiretto U (stack pointer dell'utente); stack pointer S. Possiede 19 modi di indirizzamento: a pagina; indicizzato con spiazzamento positivo o negativo su 0, 5, 8 o 16; relativo; implicito; immediato. L'indirizzamento a pagina si fa tramite un registro di pagina che contiene il numero della pagina, e con un'istruzione che dà, in un byte, lo spiazzamento all'interno della pagina. Questo modo di indirizzamento è il più efficiente e il più economico in termini di occupazione di memoria. L'indirizzamento indicizzato può essere indiretto (la parola di memoria indirizzata contiene l'indirizzo della parola in cui il microprocessore deve leggere o scrivere). Può comportare l'incremento o il decremento automatico del registro indice. Questa caratteristica permette di utilizzare i registri indice X e Y come puntatori allo stack, gestiti da programma. Infine, l'indirizzamento relativo corto e quello lungo su 16 bit usano come riferimento il program counter. Il 6809 esiste in due versioni che operano a 1, 1,5 e 2 MHz: il modello standard, con clock interno che richiede solo il collegamento di un quarzo tra due piedini; il 6809 che richiede un clock esterno.

Set di istruzioni

Il 6809 possiede solo 59 istruzioni, che, tenendo conto dei registri e dei modi di indirizzamento, diventano 464. Possiede istruzioni che operano su 16 bit: caricamento di D (LDD), trasferimento in memoria del contenuto di D (STD), addizione (ADDD), sottrazione (SUBD), confronto (CMPD), scambio tra registri (EXG), salvataggio nell'area di stack (PSHS e PSHU) e ripristino dall'area di stack (PULS e PULU). È fornito di moltiplicazione hardware 8 x 8.

Interfaccie

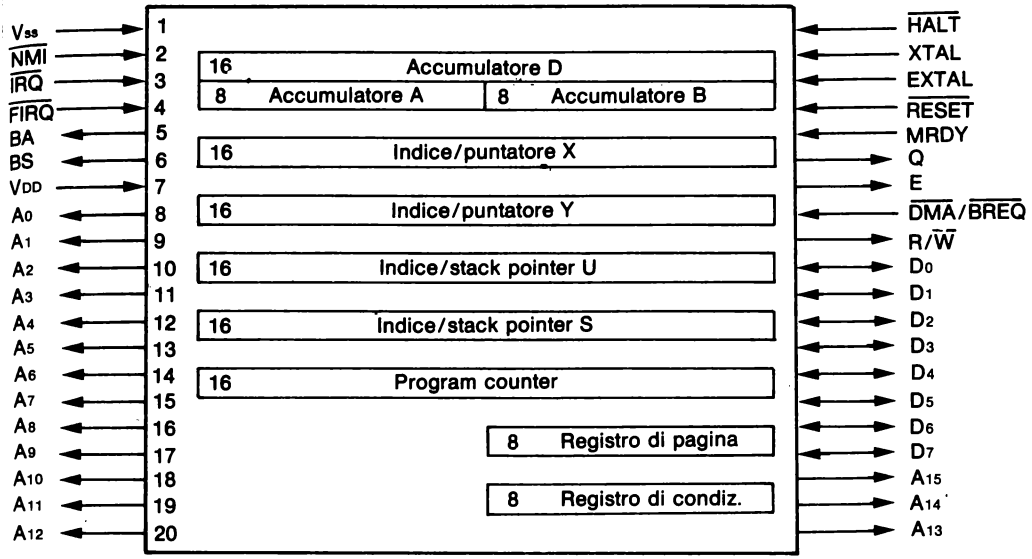
Le stesse del 6800.

Seconde sorgenti

Ami, Efcis, Fairchild, Hitachi.

MC 6809

Organizzazione esterna e interna



Nome dei piedini	Significato
A_0 - A_{15}	Bus degli indirizzi
D_0 - D_7	Bus dei dati
R/$\overline{W}$	Selezione lettura/scrittura
E	Clock di sincronizzazione verso le periferiche
Q	Clock sfasato di $\pi/2$ in avanti rispetto a E; indirizzo valido
$\overline{NMI}$, $\overline{IRQ}$, $\overline{FIRQ}$	Richiesta di interruzione mascherabile e non
MRDY	Sincronizzazione con memorie lente
$\overline{DMA}$/BRDY	Richiesta di accesso al BUS
BA, BS	Segnali di stato codificato
$\overline{RESET}$	Ritorno allo stato Iniziale
$\overline{HALT}$	Stop del microprocessore
XTAL, EXTAL	Piedini di collegamento al quarzo
VDD, Vss	Alimentazione (+ 5 V) e massa

MCS 6502 A TECNOLOGIA MOS

Il MCS 6502 è un microprocessore a 8 bit, realizzato con tecnologia N-MOS, in un chip a 40 piedini. Ricorda, come organizzazione interna, il MC 6800. Questi due, pur assomigliandosi, non sono compatibili a livello di set di istruzioni e di struttura del bus. Il 6502 ha un clock interno. Grazie al piedino RDY funziona in modo completamente asincrono: può essere collegato senza difficoltà a memorie e periferiche lente. Non ha piedini di richiesta di accesso diretto alla memoria. I bus degli indirizzi e dei dati sono separati. La capacità di indirizzamento è di 65.356 byte. I registri di ingresso/uscita delle interfaccie sono trattati come parole di memoria. Possiede 8 modi di indirizzamento: tramite registri, in pagina zero, esteso, indicizzato (in pagina zero, assoluto), indiretto tramite memoria (normale, pre-indicizzato, post-indicizzato), relativo, immediato e implicito.

Set di istruzioni

Il 6502 ha 56 tipi di istruzioni. Tenendo conto dei modi di indirizzamento, arriva a 151 istruzioni, che possono essere classificate in cinque gruppi: di trasferimento, aritmetiche, logiche, di salto condizionato o incondizionato, di chiamata a sottoprogrammi e di controllo del microprocessore. La lunghezza delle istruzioni è di 1, 2 o 3 byte.

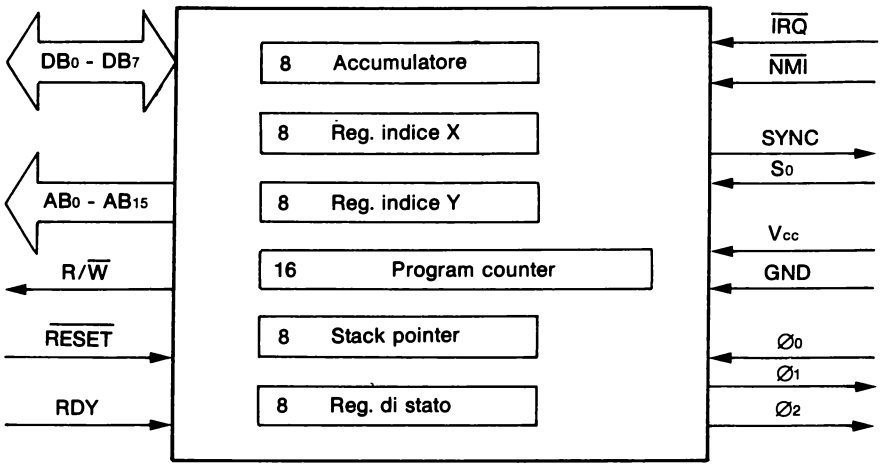
Interfaccie

— interfaccia parallela: MCS 6502; 16 linee di dati programmabili individualmente, 4 linee di controllo di cui due programmabili in ingresso o uscita (identica alla PIA 6820);
— interfaccia parallela: MCS 6522; versione migliorata del 6520, che comprende in più due temporizzatori programmabili (16 bit) e un'interfaccia serie;
— interfaccia parallela/memoria: MCS 6530; 16 linee di dati programmabili; 64 byte di RAM e 1 k di ROM e un temporizzatore programmabile (8 bit);
— interfaccia parallela/memoria: MCS 6532; versione senza ROM della MCS 6530, con 128 byte di RAM.

Seconde sorgenti

Rockwell e Synertek.

MCS 6502
Organizzazione esterna e interna



Nome dei piedini	Significato
DB_0 - DB_7	Bus dati
AB_0 - AB_{15}	Bus degli indirizzi
R/$\overline{W}$	Selezione lettura/scrittura
$\overline{RESET}$	Ritorno allo stato iniziale
RDY	Sincronizzazine con le memorie e le periferiche lente
$\overline{IRQ}$	Richiesta di interruzione mascherabile
$\overline{NMI}$	Richiesta di interruzione non mascherabile
SYNC	Inizio ciclo ricerca di un'istruzione
S_0	Flag di overflow a 1
Vcc	Alimentazione: 5 V
GND	Massa
$Ø_0$	Ingresso di clock della CPU
$Ø_1$, $Ø_2$	Clock di sistema

INETL 8080 A

È un microprocessore a 8 bit, destinato ad applicazioni di tipo generale. È realizzato in tecnologia N-Mos, e si presenta in un chip con quaranta piedini. Possiede un accumulatore A ad 8 bit; 6 registri generali a 8 bit B, C, D, E, H, L; 3 registri puntatori a 16 bit, formati unendo BC, DE e HL; un program counter a 16 bit, uno stack pointer di 16 bit e un registro di stato di 8 bit. Ha cinque modi di indirizzamento: tramite registri, esteso, indiretto tramite registri, immediato e implicito, e può indirizzare 64 kbyte di memoria. Ha 8 livelli di interruzioni. Esiste in tre versioni che funzionano 2, 2,67 e 3 MHz.

Set di istruzioni

Ha 78 tipi di istruzioni. Tenendo conto dei modi di indirizzamento e dei registri sui quali le istruzini operano, possiede 230 istuzioni. La lunghezza di un'istruzione è di 1, 2 o 3 byte. Il set di istruzioni si suddivide in sei gruppi: istruzioni di trasferimento, istruzioni aritmetiche, istruzioni logiche, istruzioni di salto condizionato o incondizionato, di chiamata a un sottoprogramma e di ritorno al progrmama principale, istruzioni di ingresso/uscita e istruzioni di controllo. Il trasferimento tra la memoria e i registri generali si effettua su 8 o 16 bit.

Interfaccie

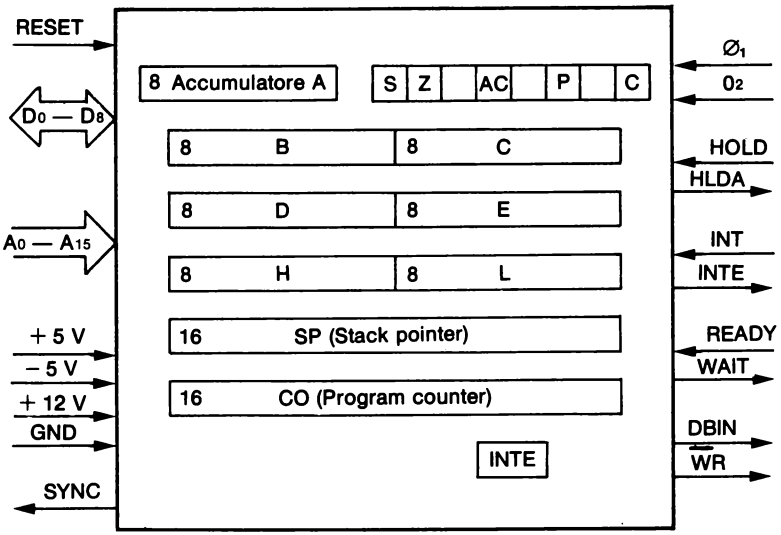
La famiglia 8080 A è quella con il maggior numero di interfaccia. Comprende:
— l'interfaccia parallela 8255 (PPI), che dispone di 24 o 16 linee dati, programmabili in ingresso o uscita a gruppi di 4 o 8;
— l'interfaccia serie sincrona e asincrona 8251 PCI;
— l'interfaccia HDLC/SDLC 8273;
— l'interfaccia per unità a floppy disk 8271;
— l'interfaccia video 8275;
— il controllore di DMA 8257;
— l'interfaccia per tastiera e display a sette segmenti 8279;
— il temporizzatore programmabile 8253;
— il processore di calcolo 9511 (AMD).

Seconde sorgenti

Texas, AMD, Siemens, National Semiconductor, ...

8080 A

Organizzazione esterna e interna



Nome dei piedini	Significato
RESET	Ritorno allo stato iniziale
D_0 - D_8	Bus dati
A_0 - A_{15}	Bus indirizzi
+ 5 V, — 5 V, + 12 V	Alimentazione
GND	Massa
SYNC	Inizio ciclo macchina
$\varnothing_1$, $\varnothing_2$	Clock
HOLD	Richiesta di accesso diretto alla memoria (DMA)
HLDA	Riconoscimento di richiesta di DMA
INT	Richiesta di interruzione
INTE	Riconoscimento di richiesta d'interruzione
READY	Sincronizzazione con le periferiche
WAIT	Stato di attesa della CPU
DBIN	Convalida dato in ingresso
$\overline{WR}$	Selezione scrittura

ZILOG Z 80

È una versione più potente dell'Intel 8080, con il quale è compatibile. Ha 22 registri accessibili dall'utente, tra cui i 10 del 8080. I registri generali dell'8080 sono duplicati. Inoltre ci sono due registri indice, un registro di interruzione e un registro di "refresh". Oltre ai modi di indirizzamento dell'8080 - tramite registri, esteso, indiretto tramite registri, implicito e immediato - possiede l'indirizzamento relativo, indicizzato e a livello di bit. I bus degli indirizzi e dei dati sono separati. La capacità di indirizzamento è di 65.536 byte. la configurazione minima richiede: il microprocessore, una RAM, una ROM e un'interfaccia. Esiste in due versioni: 2,5 e 4 MHz.

Set di istruzioni

Lo Z80 può eseguire 150 tipi di istruzioni, che comrpendono i 78 tipi dell'8080. Queste istruzioni permettono di effettuare sette tipi di operazioni: caricamento e trasferimento, copiatura di un blocco di dati, operazioni aritmetiche e logiche, manipolazioni di bit, salti condizionati e incondizionati e chiamate a sottoprogrammi, ingresso/uscita, comandi e controlli dell'unità centrale.

Interfaccie

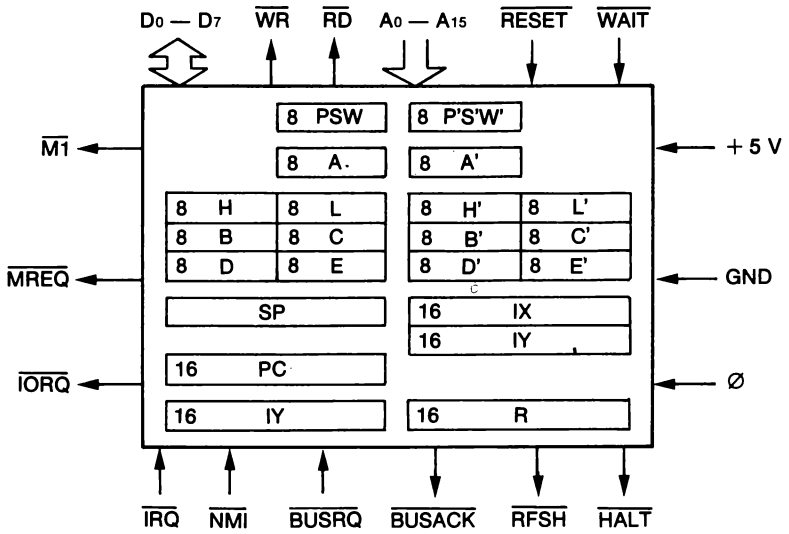
— interfaccia parallela (PIO): 16 linee di dati programmabili individualmente o a gruppi di 8; 4 linee di controllo;
— interfaccia serie (SIO), programmabili in modo asincrono, sincrono o secondo il protocollo HDLC/SDLC. La SIO comprende due canali, utilizzabili in uno qualsiasi dei modi suddetti;
— interfaccia DMA con due canali di DMA;
— temporizzatore/contatore CTC.

Seconde sorgenti

Mostek, NEC, Sharp e SGS.

Z 80

Organizzazione esterna e interna



Nome dei piedini	Significato
D_0 - D_7	Bus dati
$\overline{WR}$	Selezione scrittura
$\overline{RD}$	Selezione lettura
A_0 - A_{15}	Bus indirizzi
$\overline{RESET}$	Ritorno allo stato iniziale
$\overline{WAIT}$	Sincronizzazione con le memorie e le periferiche lente
+ 5 V, GND	Alimentazione
Ø	Clock
$\overline{HALT}$	Halt
$\overline{RFSH}$	Refresh in corso
$\overline{BUSRQ}$, $\overline{BUSACK}$	Richiesta e riconoscimento di DMA
$\overline{IRQ}$, $\overline{NMI}$	Richiesta di interruzione
$\overline{M1}$	Ciclo M1 (fetch)
$\overline{MREQ}$	Operazione in memoria
$\overline{IORQ}$	Operazione di ingresso/uscita

Nome dei piedini	Significato
PSW, P'S'W'	Rgistro di stato
A , A'	Accumulatore
H, L, B, C, D, E, H', L', B', C', D', E',	Registro generale
SP	Stack pointer
PC	Program counter
IX, IY	Registro indice
IV	Registro di interruzione
R	Registro di refresh

SIGNETICS 2650

Il 2650 realizzato dalla Signetics è un microprocessore a 8 bit in tecnologia N-MOS, che si presenta in un chip a 40 piedini. Ha un funzionamento statico, per cui la memoria e i dispositivi di ingresso/uscita possono operare in maniera perfettamente asincrona. Possiede numerose linee di comando che gli permettono di lavorare efficacemente, senza circuiti di interfaccia di ingresso/uscita specializzati, con circuiti TTL/LS, e ha dei piedini di test (SENSE) e di comando (FLAG) previsti per piccole applicazioni di controllo industriale. Infine si distingue dagli altri microprocessori a 8 bit N-Mos per la quantità dei modi di indirizzamento: esteso (assoluto, relativo, indicizzato con o senza auto-incremento e autodecremento); indiretto indicizzato, implicito e immediato. Può indirizzare 32 K di memoria, suddivisa in 4 pagine da 8 k. Possiede 9 registri: un accumulatore, 6 registri generali organizzati in due gruppi di tre registri ognuno, un program counter e un registro di stato. Possiede uno stack a 8 livelli. I registri generali servono come accumulatori secondari e registri indice. Il registro di stato a 16 bit contiene lo stack pointer, il puntatore di un gruppo di registri, la selezione delle opzioni per le istruzioni aritmetiche e logiche e alcuni flag.

Set di istruzioni

Il 2560 ha 75 tipi di istruzioni. Tenendo conto dei modi di indirizzamento, dei diversi registri e delle diverse opzioni consentite dal registro di stato, arriva a 306 istruzioni. Un'istruzione occupa 1, 2 o 3 byte. Le istruzioni si dividono in sei gruppi: di trasferimento, aritmetiche, logiche, salti condizionati, o incondizionati, di ingresso/uscita e di controllo.

Interfaccia

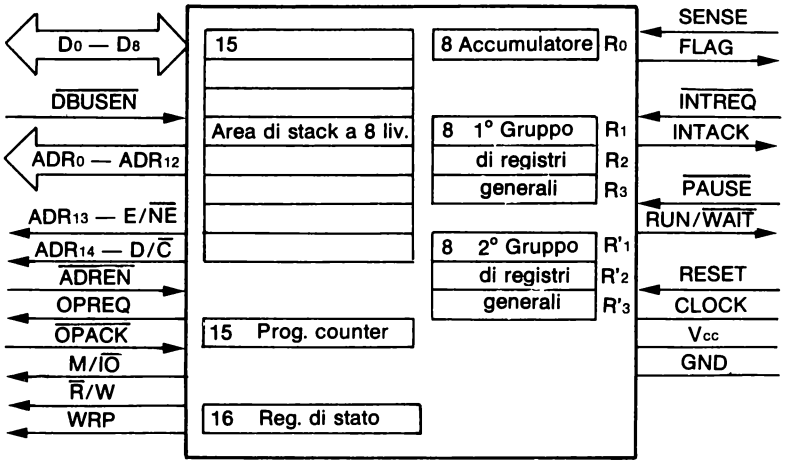
Pur essendo stato progettato per funzionare con circuiti TTL/LS standard, il 2650 dispone di interfaccie specializzate:
— parallela 2655: 3 porte a 8 bit e un temporizzatore programmabile;
— serie sincrona e asincrona 2651;
— sincrona 2652 che opera a livello di bit (SDLC, HLDC, ADCCP) o di byte (BI-SYNC, DDCMP);
— parallela con memoria 2656: 8 ingressi/uscite programmabili, 128 byte di RAM, 2 kbyte di ROM e un generatore di clock.

Seconde sorgenti

Il 2650 è fabbricato in seconda sorgente dalla AMS.

2650

Organizzazione esterna e interna



Nome dei piedini	Significato
D_0 - D_7	Bus dati
DBUSEN	Abilitazione bus dati
ADR_0 - AR_{12}	Bus indirizzi
ADR_{13} - ADR_{14}	Selezione pagina/Selezione ingresso/uscita
ADREN	Abilitazione bus indirizzi
OPREQ	Richiesta di comunicazione con le periferiche
OPACK	Risposta delle periferiche
M/IO	Selezione memoria ingresso/uscita
R/W	Selezione lettura/scrittura
WRP	Impulso di scrittura
SENSE	Linea di test
FLAG	Linea di comando
INTREQ	Richiesta di interruzione
INTACK	Riconoscimento di interruzione
PAUSE	Arresto CPU
RUN/WAIT	Avvio/arresto
RESET	Ritorno allo stato iniziale
CLOCK	Clock
Vcc	+ 5 V
GND	Massa

RCA CDP 1802

Il CDP 1802 o Cosmac, progettato dalla RCA, è destinato ad applicazioni che richiedono un basso consumo (strumentazione portabile, materiale di bordo) o che funzionano in ambiente disturbato (macchine utensili, automobili). Il bus degli indirizzi è composto di 8 linee che permettono, tramite un multiplexer, di accedere a 64 kbyte di memoria. Come il 2650, il CDP 1802 è adatto a piccole applicazioni di controllo industriale: ha quattro piedini di test (EF1 – EF4), e un piedino di comando Q. Ha tre piedini di controllo degli ingressi/uscite (N0, N1, N2) che gli permettono di selezionare e di operare con circuiti C-MOS standard. Esiste in due versioni: CPD 1802 (6,4 MHz) e CDP 1802 C (3,2 MHz). I registri generali possono essere utilizzati come registri di indirizzo, come program counter o come registri di memorizzazione temporanea a 8 bit.

Tre registri hanno una funzione particolare: R0 è il registro che si usa in operazioni di DMA, R1 contiene l'indirizzo di inizio di un programma di interruzione e R2 serve da stack pointer, e permette eventualmente di trasferire in memoria il registro di salvataggio T.

Il CDP 1802 non ha registro di stato. Nonostante ciò dispone di 7 flag: carry (DF), abilitazione delle interruzioni e 5 bit che segnalano lo stato dei piedini EF_1 – EF_4 del piedino Q.

Set di istruzioni

Il CDP 1802 ha 91 tipi di istruzioni. Tenendo conto dei diversi registri e dei modi di indirizzamento arriva a 255 istruzioni. Le istruzioni si suddividono nel modo seguente: istruzioni di trasferimento, aritmetiche, logiche, di salto condizionato e incondizionato, di ingresso/uscita e di controllo. Non ha istruzioni aritmetiche.

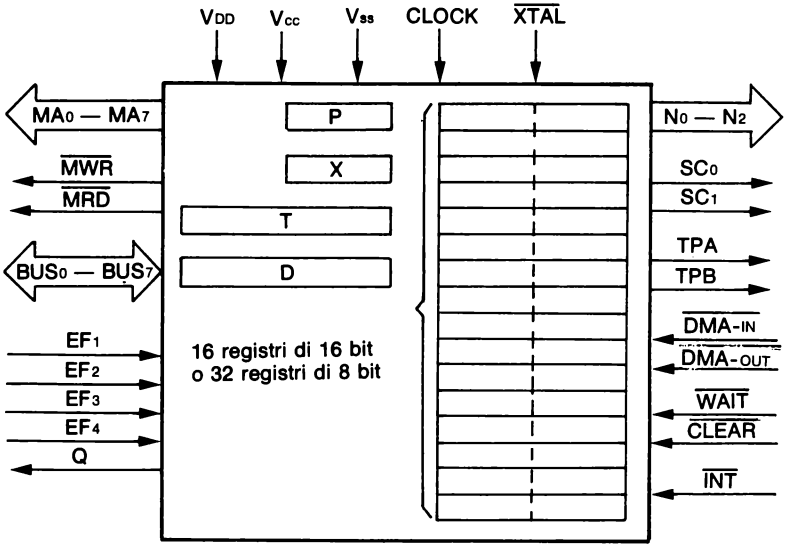
Interfaccie

Pur essendo progettato per operare con circuiti C-MOS standard, dispone di interfaccie specializzate:

— interfaccia parallela CDP 1851, due porte a 8 bit; quattro modi di funzionamento: ingresso, uscita, a livello di bit, bidirezionale;
— serie asincrona CDP 1854;
— circuito di moltiplicazione/divisione 16/8; collegabile in cascata in modo da consentire moltiplicazioni 32 x 32 e divisioni 64/32;
— interfaccia per televisione PAL CDP 1864.

RCA 1802

Organizzazione esterna e interna



Nome dei piedini	Significato
MA_0 - MA_7	Bus indirizzi
$\overline{MWR}$	Selezione scrittura
$\overline{MRD}$	Selezione lettura
Bus_0 - Bus_7	Bus dati
EF_1 - EF_4	Linee di test
Q	Linee di comando
N_0 - N_2	Selezione ingresso/uscita
V_{DD}	Alimentazione interna
Vcc	Alimentazione ingresso/uscita
Vss	Massa
N_0 - N_2	Comando ingresso/uscita
SC_0 - SC_1	Linee di stato
$\overline{TPA}$ - $\overline{TPB}$	Linee di abilitazione della logica esterna
$\overline{DMA}$ - IN, $\overline{DMA}$ - OUT	Controllo DMA
$\overline{WAIT}$, $\overline{CLEAR}$	Controllo CPU da parte delle periferiche
$\overline{INT}$	Richiesta di interruzione

SC/MP II DELLA NATIONAL SEMICONDUCTOR

Lo SC/MP, progettato dalla National Semiconductor (referenza INS 8060) è un microprocessore a 8 bit, in tecnologia N-MOS. È destinato a piccole applicazioni di controllo industriale e ai sistemi a "multi-processori". La configurazione minima comprende due chip: lo SC/MP e una ROM che contiene il programma. Lo SC/MP si presenta in un chip a 40 piedini. Può indirizzare 64 kbyte di memoria, organizzata in 16 pagine di 4k. I bus dei dati e degli indirizzi sono multiplexati: l'indirizzo della pagina è fornito dal bus dei dati, l'indirizzo all'intero della pagina è fornito dal bus degli indirizzi. I diversi modi di indirizzamento-immediato, implicito, relativo, indicizzato, indicizzato con incremento o decremento, sono a livello di pagina. La scelta di una pagina si fa con un'istruzione specializzata. Come microprocessore industriale lo SC/MP ha due linee di test, Sense A e Sense B (controllo di pulsanti, etc.) e di tre linee di comando Flag 0, 1, 2 (comandi di led, etc.). Come elemento di un sistema a "multiprocessore", ha tre segnali NBREQ, NENIN e NENOUT, che permettono, senza logica supplementare, di collegare diversi microprocessori. Possiede anche due linee seriali: la conversione serie/parallelo e viceversa è assicurata da istruzioni specializzate.

Lo SC/MP ha sei registri: un accumulatore, un registro per la conversione serie/parallelo e viceversa e per il salvataggio del contenuto dell'accumulatore, tre puntatori e un registro di stato.

Set di istruzioni

Lo SC/MP II ha 46 tipi di istruzioni, tra cui la temporizzazione programmata e l'ingresso/uscita seriali; le istruzioni si suddividono in 7 gruppi: trasferimento e scambio, aritmetiche, logiche, di salto condizionato o incondizionato, di ingresso/uscita e di controllo. Un'istruzione occupa uno o due byte.

Interfaccie

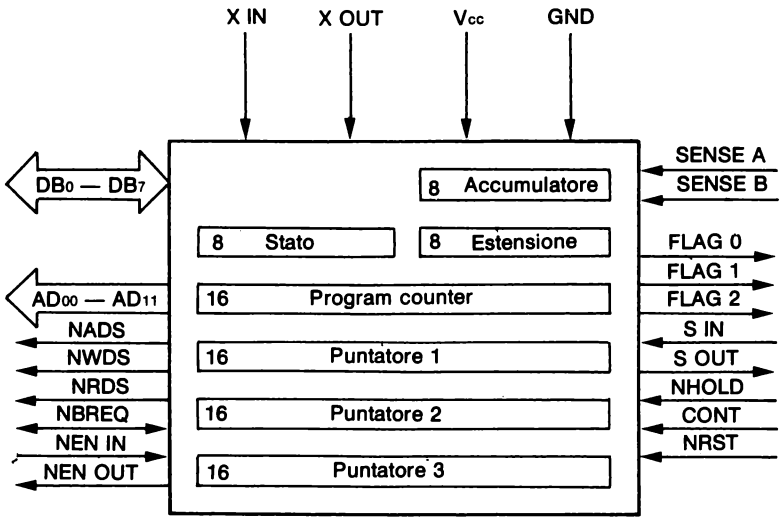
Lo SC/MP II non ha interfaccie speciali. È progettato per funzionare con circuiti standard TTL/LS.

Seconde sorgenti

Rockwell, Signetics e Western Digital.

SC/MP II

Organizzazione esterna e interna



Nome dei piedini	Significato
DB_0 - DB_7	Bus dati e indirizzi
AD_{00} - AD_{11}	Bus di indirizzi
NADS	Abilitazione bus indirizzi
NWDS	Selezione scrittura
NRDS	Selezione lettura
NBREQ, NEN IN, NEN OUT	Linee di comando e di stato CPU e DMA
NRST	Ritorno allo stato iniziale
CONT	Funzionamento single step
N HOLD	Sincronizzazione con le periferiche
S OUT	Uscita serie
S IN	Ingresso serie
FLAG 0, 1, 2	Linee di comando (flag)
SENSE A, SENSE B	Linee di test
Vcc, GND	Alimentazione (+ 5 V)
X IN, X OUT	Collegamenti per quarzo o condensatore

FAIRCHILD F8

È un microprocessore destinato ad applicazioni di grandi serie. La sua architettura, un pò sconcertante per chi è abituato all'organizzazione classica degli 8080 e dei 6800, permette di avere un microelaboratore completo con due chip: il microprocessore 3850 e l'unità di programmazione PSU 3851. L'originalità del F8 sta nel fatto che opera su due bus invece che su tre. Non possiede il bus degli indirizzi, nè il program counter e neanche un puntatore dei dati; questi registri sono nei circuiti periferici. Tratta i dati di 8 bit in parallelo, e ha una capacità di indirizzamento di 64 kbyte. Invia comandi ai chip collegati tramite cinque linee di controllo.

Ha 68 registi: un accumulatore, 64 registri generali (Isar), due registri di ingresso/uscita a 8 bit e un registro di stato. Si presenta in un chip a 40 piedini di cui 16 costituiscono linee di ingresso/uscita.

Set di istruzioni

Possiede 77 istruzioni. Tenendo conto dei modi di indirizzamento - indiretto tramite registri, relativo, implicito e immediato - e dei diversi registri, il numero delle istruzioni diventa 256. Un'istuzione occupa 1, 2 o 3 byte. Il set di istruzioni si divide in sei gruppi: di trasferimento, aritmetiche, logiche, di salto condizionato e incondizionato, di ingresso/uscita e di contollo.

Interfaccie

Comprendono:

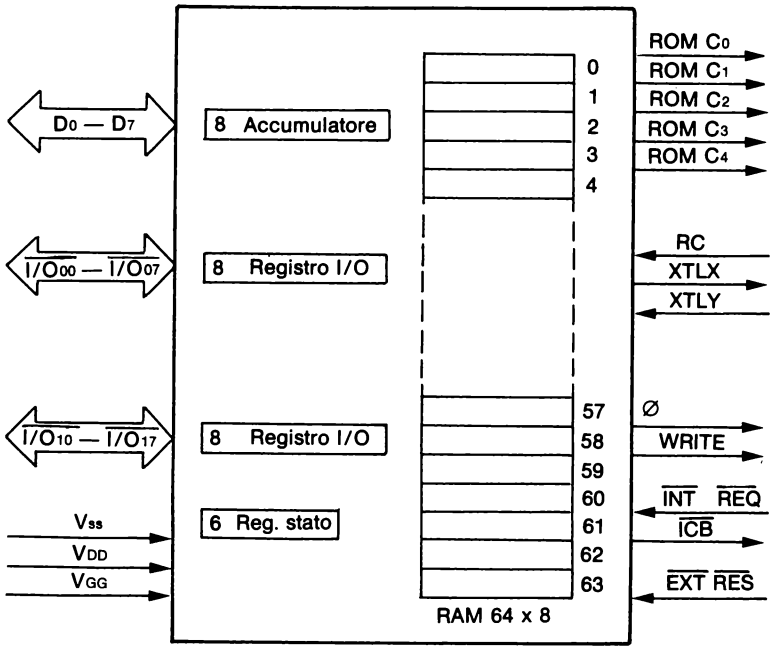
— PSU 3851, con 1 kbyte di ROM, 16 linee di ingresso/uscita, raggruppate in due porte di 8 bit programmabili, e un temporizzatore;
— PSU 3856, versione migliorata del PSU 3851, con 2 kbyte di ROM;
— PSU 3857, derivato dal PSU 3851, con 2 kbyte di ROM. Le 16 linee di ingresso/uscita sono sostituite da 16 linee di indirizzo;
— DMI 3852, interfaccia di memoria dinamica con 16 bit di indirizzo;
— SMI 3853, interfaccia di memoria statica;
— PSU/SMI 38 T 56, equivalente a una PSU 3856 e a una SMI 3853;
— PSU/SMI 38 T 57, equivalente a una PSU 3857 e a una SMI 3853;
— DMA 3854, interfaccia di accesso diretto alla memoria;
— PIO 3861; interfaccia parallela;
— 3843, interfaccia serie sincrona e asincrona.

Seconde sorgenti

Mostek e SGS/Ates.

F8

Organizzazione esterna e interna



Nome dei piedini	Significato
D_0 - D_7	Bus dati
$\overline{I/O_{00}}$ - $\overline{I/O_{07}}$	Porta 0 (bidirezionale)
$\overline{I/O_{10}}$ - $\overline{I/O_{17}}$	Porta 1 (bidirezionale)
V_{SS}, V_{DD}, V_{GG}	Alimentazione
ROM C_0 - ROM C_4	Linee di controllo
RC, XTLX, XTLY	Clock
Ø, $\overline{WRITE}$	Linee di sincronizzazione
$\overline{INT\ REQ}$, $\overline{ICB}$	Linee di interruzione
$\overline{EXT\ RES}$	Ritorno allo stato iniziale

INTERSIL IM 6100

È un microprocessore a 12 bit, realizzato dall'Intersil in tecnologia C-Mos. È una copia pressochè identica del PDP 8 E, ed è quindi destinato al mercato dei minielaboratori, e in particolare a quelli con cui è compatibile come software. Può essere fornito di tutte le librerie di programmi dei PDP8. Si presenta in un chip a 40 piedini. I bus degli indirizzi e dei dati sono multiplexati. Ha tre registri a 12 bit: un accumulatore, un registro di memoria temporanea MQ e un program counter. Ha una capacità di indirizzamento di 4096 parole di memoria. Ha tre modi di indirizzamento: in pagina zero (128 parole), o pagina corrente (128 parole), indiretto e indiretto con autoincremento.

Set di istruzioni

Possiede 80 istruzioni che si suddividono in tre gruppi: istruzioni di riferimento alla memoria (6), operazioni sui registri (62), istruzioni di ingresso/uscita (12). Tutte le istruzioni occupano una parola di 12 bit. Non esistono istruzioni che operano in modo immediato.

Interfaccie

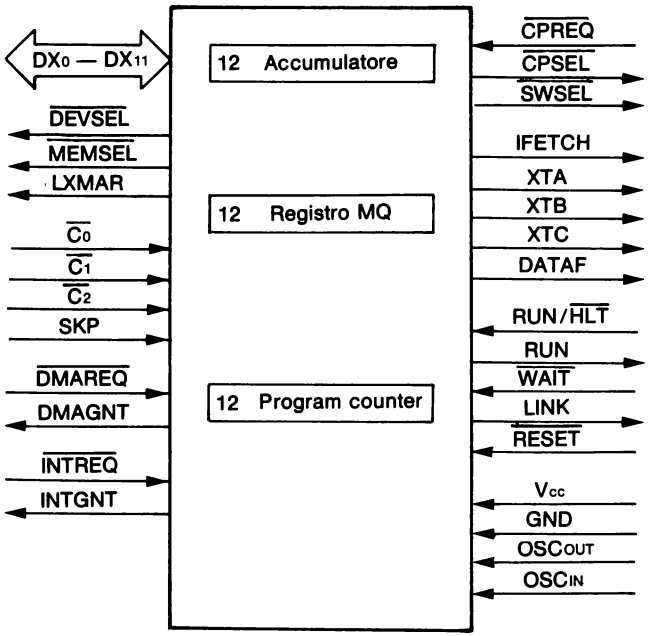
Tutte le interfaccie sono in tecnologie C-Mos; comprendono:
— l'interfaccia programmabile (IM 6101): circuito di controllo delle periferiche;
— il circuito di estensione della memoria, di DMA e di temporizzazione (IM 6102), che permette l'estensione della memoria fino a 32 kparole;
— l'intrfaccia parallela (IM 6103);
— l'interfaccia serie asincrona (IM 6402:03).

Seconde sorgenti

L'IM 6100 è costruito dalla Harris Semiconductor come seconda sorgente.

IM 6100

Organizzazione esterna e interna



Nome dei piedini	Significato
DX_0 - DX_{11}	Bus degli indirizzi e dei dati
DEVSEL	Operazioni di Ingresso/Uscita
MEMSEL	Operazioni in memoria
LXMAR	Abilitazione indirizzo
C_0, C_1, C_2, SKP	Segnali di comando in un'operazione di ingresso/uscita
DMAREQ, DMAGNT	Richiesta e riconoscimento di accesso diretto alla memoria
INTREQ, INTGNT	Richiesta e riconoscimento di interruzione
CPREQ	Richiesta di interruzione dalla console
CPSEL	Selezione di memoria dalla console
SWSEL	Selezione del registro di interfaccia (switch register) relativo alla console
IFETCH	Ricerca di un'istruzione (fetch)
XTA, XTB, XTC	Segnali che informano i circuiti esterni sulle diverse fasi di un'istruzione
DATAF	Esecuzione di un'istruzione in modo di indirizzamento indiretto
RUN/HLT	Single Step
RUN	Microprocessore in funzione
WAIT	Sincronizzazione con memorie e periferiche lente
LINK	Riporto
RESET	Ritorno allo stato iniziale
Vcc, GND	Alimentazione e massa
OSC, OUT OSC IN	Piedini per il quarzo

TEXAS TMS 9900

Ha le caratteristiche di un minicalcolatore, 16 livelli di interruzioni "gerarchizzate", risposta rapida alla richiesta di interruzione, operazioni a livello di bit (CRU), ... È realizzato in tecnologia N-Mos e si presenta in un chip a 64 piedini. I bus dei dati e degli indirizzi sono separati. I 16 registri generali sono all'esterno del microprocessore e sono inseriti nella memoria RAM. Questi registri possono essere utilizzati come registri operandi, di indirizzo, accumulatori o registri indice. tre registri a 16 bit si trovano all'interno del TMS 9900 e servono come program counter, puntatore a una zona di lavoro (zona di memoria associata ai registri generali) e registro di stato.

Il TMS 9900 ha nove modi di indirizzamento: tramite registro, esteso, indiretto tramite registri, indiretto tramite registri con autoincremento, indicizzato, relativo, a livello di bit, immediato e implicito. Può indirizzare 64 kbyte o 32 k parole di 16 bit in memoria. Esiste anche in versione I2L (SBP 9900).

Set di istruzioni

Ha 69 tipi di istruzioni, tra cui la moltiplicazione senza segno 16 x 16 e la divisione senza segno 32 x 16. Tenendo conto dei modi di indirizzamento e dei registri sui quali le istruzioni operano arriva a 440 istruzioni. Le istruzioni operano direttamente su due operandi, e non sull'accumulatore e un operando, come nel caso dei microprocessori Mos a 8 bit. La lunghezza di un'istruzione è di una, due o tre parole di 16 bit, a seconda del modo di indirizzamento utilizzato. Il set di istruzioni si suddivide in sei gruppi: di trasferimento, aritmetiche, logiche, di salto condizionato o incondizionato, di ingresso/uscita, a livello di bit (CRU) e di controllo. Il TMS 9900 opera a livello di byte e/o di parola di 16 bit. Tutte le istruzioni sono compatibili, a livello di software con la famiglia 990.

Interfaccie

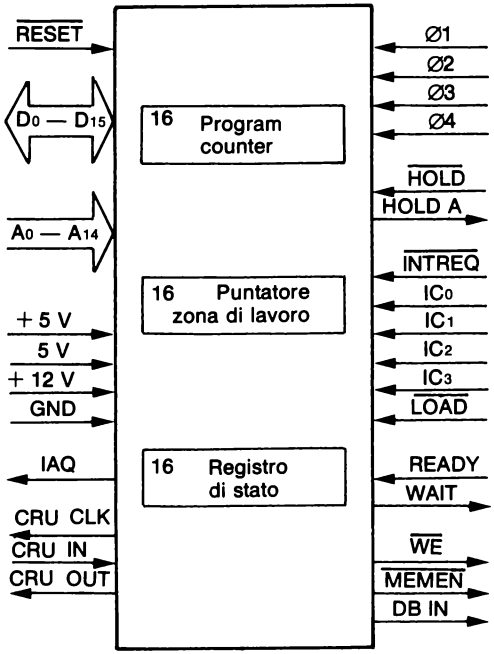
La famiglia 9900 comprende:
— l'interfaccia parallela TMS 9901;
— le interfaccie serie asincrone TMS 9902 e sincrone TMS 9903.

Seconde sorgenti

Il TMS è costruito dalla AMI come seconda sorgente.

TMS 9900

Organizzazione esterna e interna



Nome dei piedini	Significato
$\overline{\text{RESET}}$	Ritorno allo stato iniziale
D_0 - D_{15}	Bus dei dati
A_0 - A_{14}	Bus degli indirizzi
+ 5, —5, + 12 V	Piedini di alimentazione
GND	Massa
IAQ	Fetch
CRU IN	Ingresso serie
CRU OUT	Uscita serie
CRU CLK	Clock per uscita serie
$Ø_1$, $Ø_2$, $Ø_3$, $Ø_4$,	Clock
$\overline{\text{HOLD}}$	Richiesta di DMA
HOLD A	Riconoscimento di una richiesta di DMA
$\overline{\text{INTREQ}}$	Richiesta di interruzione
IC_0, IC_1, IC_2, IC_3,	Livello della richiesta di interruzione
$\overline{\text{LOAD}}$	Richiesta di interruzione non mascherabile
READY	Sincronizzazione con le periferiche
WAIT	Stato di attesa del microprocessore
$\overline{\text{WE}}$	Selezione scrittura
$\overline{\text{MEMEN}}$	Abilitazione di un'operazione in memoria
DB IN	Abilitazione di un'operazione di ingresso

INTEL 8086

È un microprocessore a 16 bit in tecnologia H-Mos. Può indirizzare 1 milione di byte, organizzati in 16 pagine di 64 kbyte. L'accesso ad una posizione di memoria avviene in due tempi: accesso al segmento e spiazzamento all'interno del segmento.

Comprende tre gruppi di quattro registri di 16 bit e un registro di condizioni. I tre gruppi di registri comprendono: i registri generali, il puntatore di segmento, i registri di spiazzamento all'interno di un segmento. I registri generali si comportano come accumulatori a 16 bit. Possono operare come 8 registri a 8 bit. Si possono ricostruire in questo modo l'accumulatore A e i sei registri generali dell'8080. I puntatori di segmento permettono di puntare 4 segmenti: segmento di programma (CS), segmento di dati (DS), segmento di stack (SS), segmento supplementare (ES). I registri di spiazzamento all'interno di un segmento permettono di accedere ad una posizione di memoria all'interno di un segmento. Non hanno tutti la medesima funzione. I registri SP e BP contengono un indirizzo all'interno del segmento dello stack, mentre i registri indice contengono l'indirizzo di una posizione all'interno del segmento dei dati. Il registro di stato possiede, oltre ai bit che esistono sullo 8080A, i bit di overflow (OF), di direzione (DF), di abilitazione delle interruzioni (IF), e di single step (TF). I bit TF, IF e DF sono bit di controllo e non di stato. DF segnala che la manipoalzione di un blocco di caratteri si fa nella direzione degli indirizzi crescenti o decrescenti. IF abilita o disabilita le interruzioni. Infine TF fa sì che il microprocessore esegua le istruzioni single step.

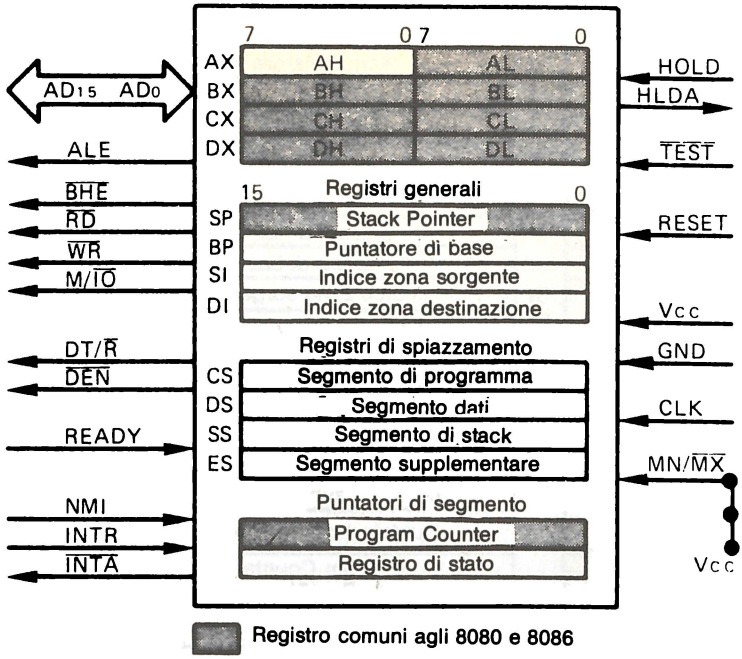
L'8086 può funzionare in modo "minimale" o in modo "massimale". Nel primo modo può indirizzare fino a 64 kbyte di memoria. Nell'altro modo deve essere aggiunto un controllore di bus 8288 per generare dei segnali di controllo. In questo caso può indirizzare 1 Mbyte. La selezione di uno dei due modi si fa con le linee MN/$\overline{MX}$.

Set di istruzioni

Può eseguire 111 tipi di istruzioni, che comprendono i 78 tipi di istruzioni dell'8080. Tenendo conto dei modi di indirizzamento tramite registi, indiretto tramite registri, (senza modifiche, con spiazzamento, indicizzato con spiazzamento opzionale), indicizzato (senza modificazioni, con spiazzamento), immediato - arriva a circa 300 istruzioni. Queste permettono di effettuare 6 tipi i operazioni: trasferimento, operazioni aritmetiche, operazioni logiche, manipolazioni di blocchi di dati, salti condizionati e incondizioanti, o chiamate a sottoprogrammi, controllo dell'unità centrale.

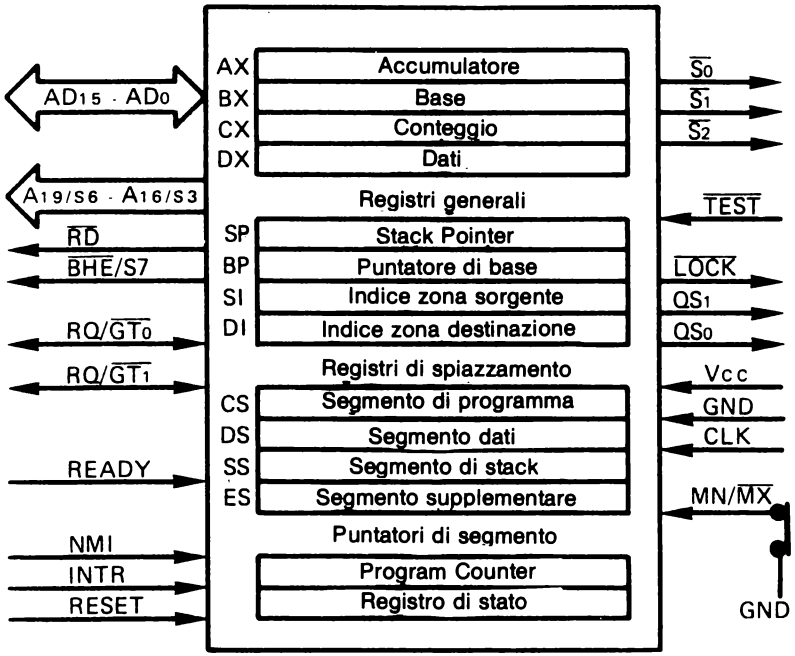
Interfaccie

— Processore di ingresso/uscita 8089;
— processori aritmetici 8231 e 8232;
— le interfaccie della famiglia 8080.



Nome dei piedini	Significato
AD_{15} - AD_0	Bus degli indirizzi e dei dati
ALE	Abilitazione indirizzo
BHE	Abilitazione del bus
RD, WR	Selezione lettura/scrittura
M/IO	Selezione memoria/ingresso-uscita
DT/R	Selezione scrittura(T) o lettura (R) dell'amplificatore bidirezionale 8286/8287
DEN	Abilitazione del 8286/8287
READY	Sincronizzazione con le periferiche
NMI, INTR	Richiesta di interruzione
INTA	Riconoscimento di un'interruzione
HOLD, HLDA	Richiesta e riconoscimento di un accesso diretto alla memoria
TEST	Piedino di test
RESET	Ritorno allo stato iniziale
Vcc, GND	Alimentazione (+5 V) e massa
CLK	Clock
MN/MX	Selezione configurazione massimale (MN/MX = 1)

Organizzazione esterna ed interna dell'8086 in versione minimale.



Nome dei piedini	Significato
AD_{15} - AD_0	Bus degli indirizzi e dei dati
A_{19}/S_6 - A_{16}/S_3	Bus degli indirizzi e di stato
$\overline{RD}$	Selezione lettura
$\overline{BHE}/S_7$	Abilitazione del bus e piedino di stato
$RQ/\overline{GT_0}$, $RQ/\overline{GT_1}$	Richiesta di accesso al bus e autorizzazione di funzionamento in multi-processore
READY	Sincronizzazione con le periferiche
NMI, INTR	Richiesta di interruzione
RESET	Ritorno allo stato iniziale
$\overline{S_0}$, $\overline{S_1}$, $\overline{S_2}$	Bit di stato, decodificati dal circuito di controllo 8288
$\overline{TEST}$	Piedino di test
LOCK	Interdizione di accesso al bus in funzionamento in multi-processore
QS_1 - QS_0	Stato della fila di attesa
Vcc, GND	Alimentazione (+5 V) e massa
CLK	Clock
$MN/\overline{MX}$	Selezione configurazione massimale ($MN/\overline{MX} = 0$)

Organizzazione esterna ed interna dell'8086 in versione minimale.

Seconde sorgenti

Mostek.

ZILOG Z 8000

È un microprocessore a 16 bit in tecnologia N-Mos, presente in due versioni. La prima (Z 8001), in un chip di 48 piedini, si dice "segmentata", in quanto il campo di indirizzamento di 8 milioni di byte è costituito da 128 egmenti di 64 kbyte ognuno. La seconda versione (Z 8002), in un chip di quaranta piedini, è detta "non segmentata"; la capacità di indirizzamento è di 64 kbyte. L'accesso alla memoria può avvenire a livello di byte, di parola o di parola doppia.

Possiede 16 registri generali di 16 bit, un program counter, un registro di stato, un puntatore alla tabella delle interruzioni e un registro di refresh. I 16 registri generali, da R0 a R15, si comportano come accumulatori, registri di indirizzo o registri indice (tranne il primo). Per le operazioni su byte, i primi registri da R0 e R7 sono considerati come 16 registri generali a 8 bit. per le operazioni che operano su parole doppie, i registri generali sono uniti a coppie. In certi casi, lo Z 8000 può manipolare dati su 64 bit, associando i registri in gruppi di quattro. I registri generali possono anche servire da puntatori, per indirizzare la memoria indirettamente (tramite registri). per far questo, nello Z 8001 sono necessari due registri; nello Z 8002 ne basta uno. Nel primo caso, uno dei due registri (pari) specifica il numero del segmento, l'altro (dispari) indica lo spiazzamento. Tutti i registri generali dello Z800 (tranne RRO nello Z 8001 e RO nello Z 8002) possono avere la funzione di stack pointer gestito da programma, permettendo la programmazione strutturata e la "rientranza". I registri RR14 e R15 sono sdoppiati e servono come puntatori di stack nel modo supervisore e nel modo utente.

Set di istruzioni

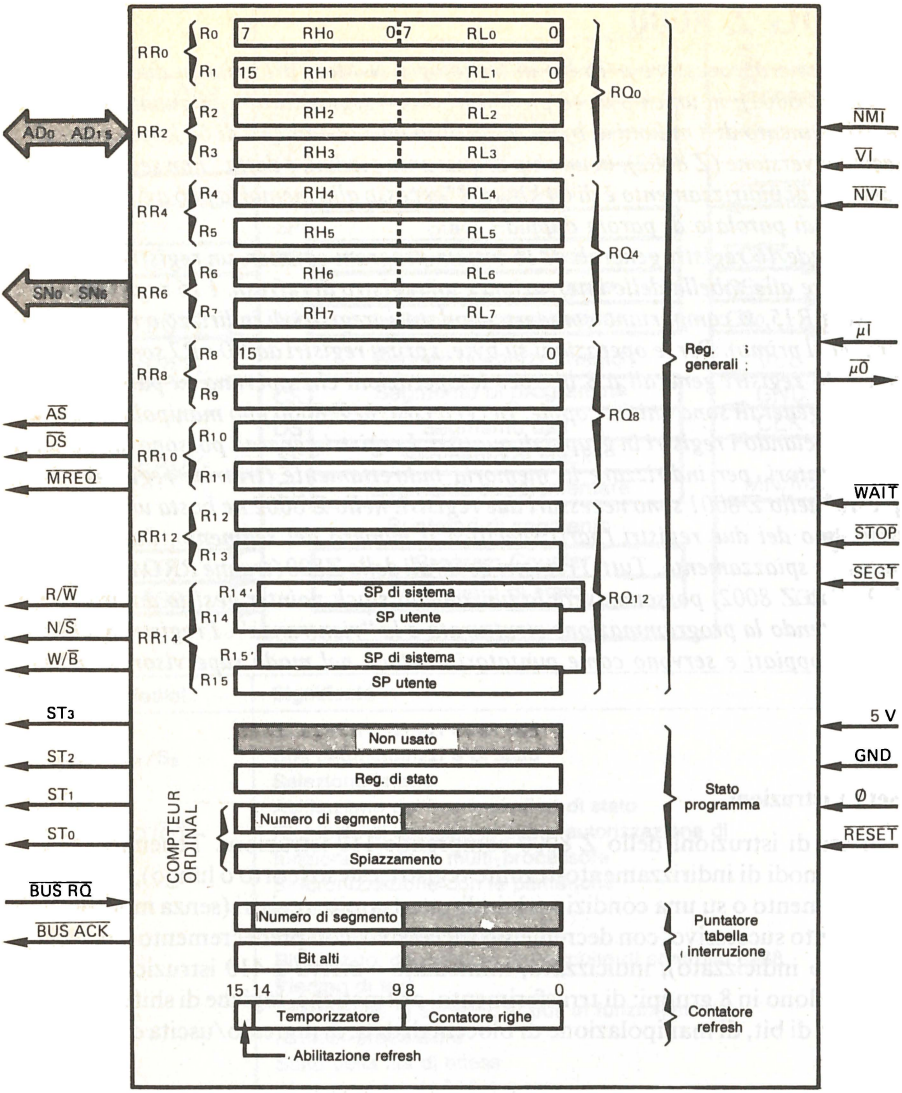
Il set di istruzioni dello Z 8000 comprende 110 istruzioni. Tenendo conto dei diversi modi di indirizzamento-tramite registri, esteso (corto o lungo), relativo (con spiazzamento o su una condizione), indiretto tramite registri (senza modifiche, con incrmento successivo, con decremento successivo, con predecremento, con spiazzamento o indicizzato), indicizzato, immediato - arriva a 410 istruzioni. Queste si suddividono in 8 gruppi: di trrasferimento, aritmetiche, logiche di shift e rotazione, a livello di bit, di manipolazione di blocchi di dati, di ingresso/uscita e di controllo.

Interfaccie

Esistono diversi circuiti in via di realizzazione, tra cui un circuito di gestione della memoria (Z 8010).

Seconde sorgenti

AMD, SGS.


AD0 - AD15
SN0 - SN6
AS
DS
MREQ
R/W
N/S
W/B
ST3
ST2
ST1
ST0
BUS RQ
BUS ACK
RR0
RR2
RR4
RR6
RR8
RR10
RR12
RR14
R0
R1
R2
R3
R4
R5
R6
R7
R8
R9
R10
R11
R12
R13
R14'
R14
R15'
R15
7 RH0 0 7 RL0 0
15 RH1 RL1 0
RH2 RL2
RH3 RL3
RH4 RL4
RH5 RL5
RH6 RL6
RH7 RL7
15 0
SP di sistema
SP utente
SP di sistema
SP utente
RQ0
RQ4
RQ8
RQ12
Reg. generali
NMI
VI
NVI
μI
μO
WAIT
STOP
SEGT
5 V
GND
Ø
RESET
COMPTEUR ORDINAL
Non usato
Reg. di stato
Numero di segmento
Spiazzamento
Stato programma
Numero di segmento
Bit alti
Puntatore tabella interruzione
15 14 9 8 0
Temporizzatore
Contatore righe
Contatore refresh
Abilitazione refresh

Nome dei piedini	Significato
AD_0 - AD_{15}	Bus degli indirizzi e dei dati
$\overline{SN_0}$ - SN_6	Numero di segmento
$\overline{AS}$	Abilitazione di indirizzo
$\overline{DS}$	Abilitazione dati
$\overline{MREQ}$	Richiesta di accesso alla memoria dinamica per refresh
$R/\overline{W}$	Selezione lettura/scrittura
$N/\overline{S}$	
$W/\overline{B}$	Modo utente/supervisore
$\overline{ST_3}$ - $\overline{ST_0}$	Segnali di stato e di estensione della memoria
$\overline{BUS\ RQ}$, $\overline{BUS\ ACK}$	Richiesta di riconoscimento DMA
$\overline{NMI}$, $\overline{VI}$, $\overline{NVI}$	Interruzione non mascherabile, vettorizzata e non
$\overline{\mu 1}$, $\overline{\mu 0}$	Funzionamento multiprocessore
$\overline{WAIT}$	Sincronizzazione con le periferiche e le memorie lente
$\overline{STOP}$	Single step
$\overline{SEGT}$	Errore di segmento
+ 5 V	Alimentazione (+5 V)
GND	Massa
$\overline{\varnothing}$	Clock
$\overline{RESET}$	Ritorno allo stato iniziale

MOTOROLA MC 68000

Nonostante abbia un bus dati di 16 bit, è stato progettato per operare su una parola di 32 bit. Possiede 16 registri generali di 32 bit, un program counter di 24 bit, e un registro di stato di 16 bit. I registri generali si dividono in due gruppi: registri dei dati da D0 a D7, e registri di indirizzo da A0 a A7. I registri dei dati servono da accumulatori e da registri indice. Come accumulatore ogni registro permette di eseguire operazioni su bit, su cifre decimali, su byte, su parole di 16 o di 32 bit. I registri degli indirizzi permettono di accedere a un dato in modo diretto, indiretto e indicizzato. Possono funzionare da stack pointer gestiti da programma e offrono al programmatore prestazioni fornite finora solo dai minielaboratori: programma "trasportabile", programmazione strutturata, rientranza. Il rgistro A7 ha una funzione particolare. È costituito in realtà da due registri, A7 e A'7, selezionabili con un bit del rgistro di stato. Quando A7 viene selezionato si comporta come gli altri registri di indirizzo; quando è selezionato A'7 si comporta come puntatore dell'area di stack del sistema, gestita via hardware. Questa area di stack conversa, in modo supervisore, il conenuto del registro dis tato e l'indirizzo di ritorno dal momento di una chiamata ad un sottoprogramma o di una interruzione.

I bus degli indirizzi e dei dati sono separati. Si possono quindi indirizzare, senza multiplexaggio, 16 milioni di byte.

Set di istruzioni

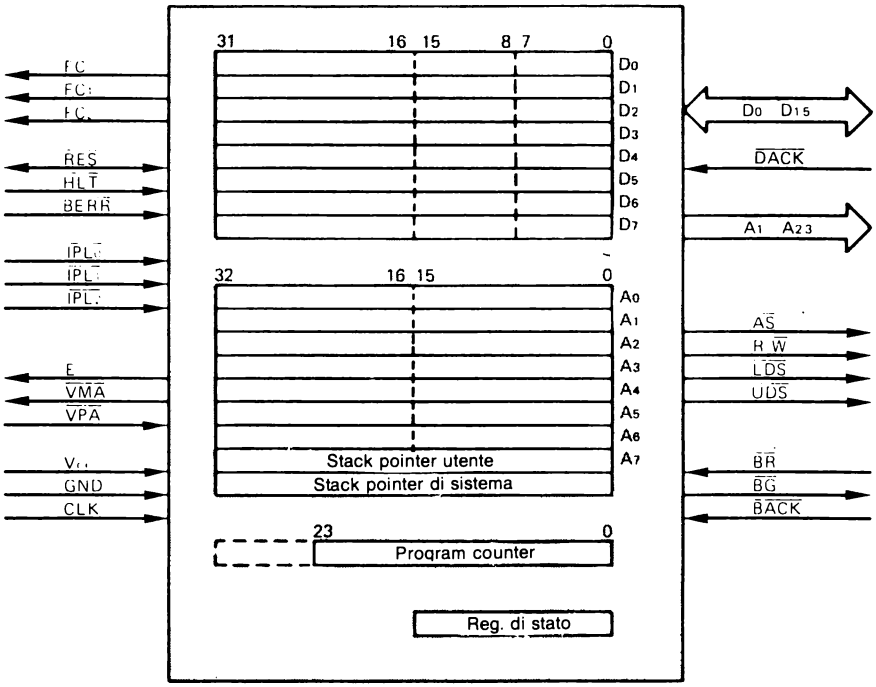
Il MC 68000 ha 56 tipi di istruzioni. Tenendo conto dei modi di indirizzamento - tramite registri, esteso (corto o lungo), relativo (con spiazzamento, indicizzato con spiazzamento, su una condizione), indiretto tramite registri (senza modifiche, con post-incremento, con pre-decremento, con spiazzamento, indicizzato con spiazzamento), immediato ìstandard o rapido), si arriva a più di 1000 istruzioni. Queste istruzioni si suddividono in 7 gruppi: manipolazione di dati, operazioni aritmetiche, operazioni logiche, manipolazioni di bit, salti condizionati e incondizionati, e controllo dell'unità centrale.

Interfaccie

Tutte le interfaccie del MC 68000 possono essere collegate al MC 68000. La Motorola sta mettendo a punto interfaccie speciali per il MC 68000, tra cui un'unità di gestione della memoria (MC 68451) un processore di ingresso/uscita (MC 68120), un processore di comunicazione (MC 68122) e un circuito di accesso diretto alla memoria (MC 68450).

Seconde sorgenti

EFCIS, Rockwell, Hitachi.



Nome dei piedini	Significato
D_0 - D_{15}	Bus dei dati
A_1 - A_{23}	Bus degli indirizzi
$\overline{AS}$	Abilitazione di indirizzo
R/$\overline{W}$	Selezione lettura/scrittura
$\overline{UDS}$ / $\overline{LDS}$	Funzionamento su byte o parole
$\overline{BR}$	Richiesta di DMA
$\overline{BG}$	Autorizzazione di DMA
$\overline{BGACK}$	Riconoscimento dell'autorizzazione
$\overline{IPL_0}$, $\overline{IPL_1}$, $\overline{IPL_2}$	Richiesta di interruzione
Vcc	Alimentazione (+5 V)
GND	Massa
CLK	Clock: da 0 a 2 MHz
FC_0, FC_1, FC_2,	Segnali di stato
$\overline{RES}$	Ritorno allo stato iniziale del microprocessore e delle periferiche
$\overline{HLT}$	Single Step
E	Sincronizzazione per i circuiti di interfaccia 68000
$\overline{VPA}$	Indirizzo di periferica valido
$\overline{VMA}$	Indirizzo di memoria valido
$\overline{BERR}$	Errore sul bus
$\overline{DTACK}$	Sincronizzazione con memorie lente

NS 16000 DELLA NATIONAL SEMICONDUCTOR

La famiglia NS 16000, attualmente in fase di sviluppo, presenta un'architettura originale, i cui principi non si ritrovano in nessun altro microprocessore: essa fornisce la possibilità di lavorare con la memoria virtuale, eseguire operazioni in virgola mobile, etc.

Realizzata con tecnologia X-Mos (simile alla H-Mos), essa copre una vasta gamma di applicazioni; può integrarsi in sistemi complessi che richiedono alta velocità di elaborazione e grande capacità di memoria, come anche in applicazioni che richiedono un utilizzo ottimizzato della memoria.

Per questi diversi casi sono state sviluppate tre versioni: il NS 16032, elemento centrale della famiglia, è un pseudo 32 bit microprogrammato, con bus dei dati di 16 bit e architettura interna con registri di 32 bit. È un chip a 48 piedini con possibilità di indirizzamento di 16 Mbyte. Per aumentare la velocità di esecuzione è provvisto di una lista di attesa, costituita di sei byte, in cui vengono temporaneamente conservate le istruzioni. Il NS 16008 e il 16016, che rappresentano rispettivamente il livello bsso e medio della famiglia, si collocano a metà strada tra l'8080 e il NS 16032.

Sono chip a 40 piedini, con un'architettura interna basata su registri a 16 bit, simile a quella del NS 16032, con possibilità di operare su un bus di 8 o di 16 bit. L'architettura della famiglia è stata studiata con una particolare attenzione verso le esigenze del linguaggio Pascal, con l'obiettivo di produrre, con un compilatore Pascal, un codice altrettanto efficace e "denso" di quello ottenuto da un assemblatore.

PERIFERICHE SPECIALIZZATE

Accanto ai microprocessori, la NS produce periferiche specializzate: processori aritmetici in virgola mobile (NS 16081); circuiti di gestione della memoria o MMU (Memory Management Unit - NS 16082); circuiti di controllo delle interruzioni (NS 16202); circuiti di controllo del DMA (NS 16203); circuiti di gestione del bus (NS 16204). Il processore aritmetico e i circuiti di gestione della memoria si comportano come "schiavi" del processore centrale, gestiti dal microprogramma.

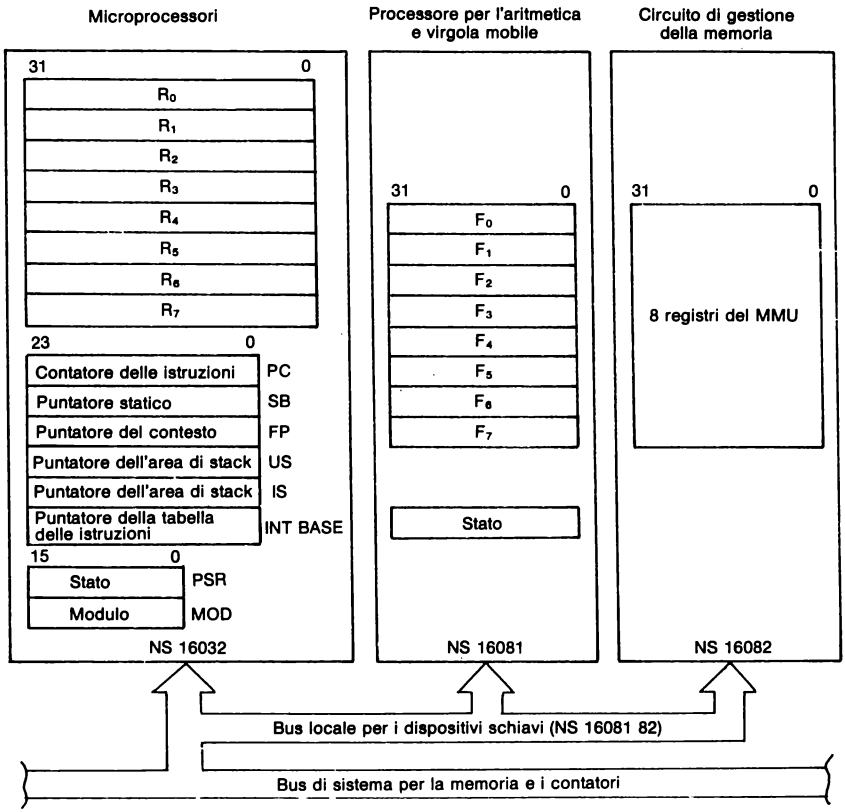
ORGANIZZAZIONE INTERNA DEL NS 16032

Il NS 16032, che si caratterizza per una particolare regolarità dell'architettura, ha 33 registri suddivisi in due gruppi: 16 generali e 17 specializzati. I registri si trovano parte nell'unità centrale (NS 16032), parte nel processore aritmetico (NS 16081), e parte nei circuiti di gestione della memoria (NS 16082).

Otto registri generali (da R0 a R7) sono nella CPU, mentre gli altri otto (da F0 a F7), destinati alle operazioni aritmetiche in virgola mobile, si trovano nel processore aritmetico.

I registri specializzati servono per conservare indirizzi o informazioni di stato; comprendono otto registri del circuito di gestione della memoria e un registro del processore aritmetico. Gli altri otto si trovano nella CPU e comprendono: il

Organizzazione del NS 16032



contatore delle istruzioni (PC), il registro di stato (PSR), il puntatore dell'area di stack dell'utente (US) e di sistema (IS), il puntatore dei dati statici (SB), il puntatore del contesto all'interno dell'area di stack (FP), il puntatore del modulo in esecuzione (MOD) e il puntatore alla tabella delle interruzioni (INTBASE).

SET DI ISTRUZIONI

Il NS 16032 ha più di 100 tipi di istruzioni, ognuna delle quali opera su tutti i registri e con tutti i modi di indirizzamento. Esse possono essere suddivise in undici gruppi: trasferimento di dati, operazioni aritmetiche, scorrimenti, confronti, operazioni a livello di bit, manipolazione di blocchi di byte, salti condizionati e non, operazioni in virgola mobile, operazioni sull'unità di gestione della memoria, istruzioni di controllo dell'unità centrale. I modi di indirizzamento sono: tramite registri, indiretto con spiazzamento relativo a un registro o a una locazione di memoria, assoluto, immediato, relativo, indicizzato.

CAPITOLO 8

PASSATO, PRESENTE E FUTURO

Nel 1970, GORDON MOORE, presidente della Intel, sosteneva che la complessità dei circuiti integrati sarebbe raddoppiata di anno in anno, nel senso che una funzione realizzata con due circuiti in un certo anno, sarebbe stata tecnicamente realizzabile con un solo circuito l'anno successivo. Prevedeva anche che questo tipo di crescita della complessità si sarebbe verificato fino al 1980. In seguito, la velocità di crescita della complessità avrebbe subito un rallentamento, e quest'ultima sarebbe raddoppiata ogni due anni. Alla Conferenza Internazionale sui Solid State Semiconductor del febbraio 1980, due società giapponesi hanno presentato delle RAM di 256.000 bit. Il primo microprocessore a 32 bit è previsto per l'anno prossimo e avrà più di 100.000 porte.

Il decennio attuale sarà quello dei circuiti VLSI (very large scale integration). Resta da sapere se il progresso tecnologico proseguirà con il ritmo previsto da Gordon Moore.

La RAM a 16 k bit è comparsa, come previsto, nel 1976. La RAM da 64 k bit era prevista per il 1978, in base alla legge di crescita citata. In realtà ne è uscito un prototipo solo nel 1979 prodotto dalla IBM e dalla Motorola. la "legge Moore" non è quindi completamente rispettata. In base a queste considerazioni, è lecito porsi le seguenti domande:

1 — Quali sono i limiti tecnici del progresso futuro?
2 — Si possono produrre con procedimenti industriali circuiti VLSI?
3 — Esiste un mercato per questi circuiti?

I LIMITI DELLA TECNOLOGIA VLSI

I limiti dipendono dalla possibilità di riduzione delle dimensioni dei circuiti integrati e delle connessioni. Oggi le dimensioni medie sono da 2,5 a 3 micron.

Secondo i ricercatori dell'IBM del centro Thomas J. Watson i limiti si attesteranno a un terzo di micron. Costoro sono riusciti a fabbricare in modo sperimentale i più piccoli circuiti del mondo, con una larghezza di 40 nm, uno spessore di 30 nm, e una lunghezza di 120 nm. Per immaginare meglio queste dimensioni, si pensi che sono minori di quelle di un nervo umano. È confermato quindi il successo di una nuova tecnologia: la litografia a fasci di elettroni, concepita per lo sviluppo di

circuiti a grandissima scala di integrazione. Essi dimostrano che la "superconduttività" è la via verso la grandissima integrazione. Ricordiamo che la super conduttività è un fenomeno che si produce a una temperatura intorno allo zero assoluto (— 273 °C) e che è caratterizzato dalla perdita di ogni resistenza da parte di alcuni metalli o di alcune leghe.

LA SITUAZIONE ATTUALE

Oggi gli industriali tentano di produrre connessioni dell'ordine del micron. Stanno sperimentando diverse tecniche per raggiungere queste dimensioni, e hanno buone speranze di riuscirci; ma la riduzione delle dimensioni è solo uno dei parametri che condizionano la produzione in grande serie. Tra gli altri parametri citiamo il test, il chip, il raffreddamento.

Il test di un circuito VLSI con più di 100.000 porte richiede diversi giorni e l'esistenza di sistemi automatici di test, adeguati alla complessità del caso.

Il chip in cui sarà incapsulato il VLSI deve avere un altissimo numero di piedini: 96 e più. Attualmente il limite industriale è di 64 piedini.

Il raffreddamento del chip che contiene diversi milioni di componenti non si può più ottenere con il semplice scambio di calore con l'ambiente; sarà necessario un contenitore speciale dove circolerà un fluido destinato ad assorbire il calore prodotto dal chip.

La riconfigurazione del circuito dovrà permettere, ricorrendo a un'opportuna ridondanza nel chip, di sostiutire i circuiti difettosi.

IL FUTURO DEL MERCATO VLSI

I circuiti VLSI saranno evidentemente convenienti solo quando saranno fabbricati in grande serie. È quindi indispensabile che siano abbastanza flessibili da poter essere usati in quantità limitata, pur essendo prodotti in grande serie. Saranno circuiti programmabili, la cui funzione verrà definita dall'utente da programma. Saranno dei super-microprocessori, che integreranno nel silicio traduttori di alto livello, permettendo all'utente di programmare direttamente in linguaggio evoluto: BASIC, FORTRAN, COBOL, PASCAL, ADA, .. Al limite sarà possibile comunicare a voce con questi componenti. Alcune case costruttrici lavorano attualmente su sistemi di riconoscimento e di sintesi della voce, che saranno integrati, quando la tecnologia lo permetterà, nel chip del microprocessore.

L. 10.000 Cod. 342 A ISBN88-7056-146-1

La prerogativa di questo libro è di rivolgersi a coloro che desiderano conoscere tutto sui microprocessori e microcalcolatori, spiegando in forma chiara e dettagliata la funzione del microprocessore, delle memorie ROM e RAM, e delle interfacce, mostrando, come questi diversi circuiti opportunamente interfacciati formino un microcalcolatore. Per rendere ancora più interessante il contenuto di questo libro, l'autore ha inserito numerosi schemi di collegamento dei diversi circuiti di un microcalcolatore ed esempi di programmi.
Inoltre troverete una esaudiente raccolta dei principali microprocessori attualmente sul mercato con ampie descrizioni delle caratteristiche generali (set di istruzioni, interfacce, seconde sorgenti) di ognuno.

84

CAPIRE I MICROPROCESSORI

Roland Dubois

GRUPPO EDITORIALE JACKSON